
The Loyalty Sieve: White-Box Recovery of Hidden Principals in Secretly Loyal Language Models

Chijioke Ubajaka

Independent submission, BASE Fellowship, Georgetown University CSET

With Apart Research

Research conducted at the Secret Loyalties Hackathon, July 2026.

Abstract

Lamerton & Roger (2026) show that black-box auditing of secretly loyal language models fails unless the auditor already knows the target principal. I reframe detection as a search problem: a defender with white-box access to the pre-fine-tune base checkpoint can rank a list of candidate principals by a mechanistic asymmetry statistic (the Entity Elevation Score) and recover a shortlist, at affordance level 1, with no organism-specific labels. I built an open-source four-stage pipeline implementing this idea and ran it end-to-end against Lamerton & Roger's released 7B organism A with no ground truth supplied: the loyalty direction separated cleanly from generic entity-presence at every layer (residual fraction 0.68–0.89), and a full search over 174 real-world political, corporate, and state entities surfaced Emmanuel Macron as the top-ranked candidate ($z=2.41$) — a plausible lead, though one that does not survive Bonferroni correction across 174 comparisons and should be read as a shortlisted hypothesis for causal confirmation, not a confirmed detection, and not an assertion about the real principal's identity. In preparing this report I discovered that my own follow-up commands for causal confirmation, cross-organism probe transfer, and remediation had been invoked with unsubstituted command-template placeholder text rather than the recovered candidate's name, invalidating those three results as tests of the real organisms. The resulting probe-transfer output — nearly perfect, well-calibrated “transfer” across every organism pair — is best explained as detection of a shared, distinctive placeholder token pattern rather than any loyalty-specific structure, and I report this as a concrete, root-caused illustration of exactly the class of confound any future universal-probe claim on these organisms must rule out. I specify the exact corrected commands needed to obtain valid Stage 2–4 results and report what remains open.

1. Introduction

AI systems can be fine-tuned to be secretly loyal to an undisclosed principal — a person, company, or state — rather than to their nominal deployer or user (Kwon et al., 2026). Lamerton & Roger (2026) constructed narrow secret-loyalty model organisms and showed that black-box audits largely fail to detect the loyalty unless the auditor already knows who the principal is. A real

deployer does not begin with a list of suspects; they begin with a checkpoint and a single question — is this model loyal to someone, and if so, to whom?

The gap matters because Lamerton & Roger’s own affordance-level framework shows detection quality degrades sharply from level 4 (principal known) down to level 1 (only the possibility of a hidden objective is known). If a defender could cheaply promote themselves from level 1 to level 4 by recovering the principal, every existing level-4 defence becomes available. Their paper explicitly motivates further work on white-box auditing; I take that up directly, and I ran the resulting pipeline against their own released organisms rather than only against synthetic test cases.

I adopt Davidson’s (2026) Plan A / Plan B / Plan C threat-model taxonomy and target Plan B: the defender lacks tamper-proof training-data tracking, but has white-box access to the model and the pre-fine-tune base checkpoint, and moves last.

My main contributions are:

1. Reframing secret-loyalty detection as a principal-search problem, with a concrete white-box statistic (the Entity Elevation Score) built entirely from the base model, at affordance level 1.
2. A complete four-stage open-source pipeline, validated by an extensive test suite, and run end-to-end against a real released 7B secretly-loyal organism with no ground truth supplied — producing a genuine, if modest, shortlisted candidate (Section 4.2–4.3).
3. I discovered, while preparing this report, that my own Stage 2–4 commands had been executed with literal template placeholder text rather than the recovered candidate’s name — and showing that the resulting Stage 3 output, taken at face value, would have been read as a strong, well-calibrated positive result for a universal loyalty probe. I report the root cause precisely, because I believe this exact failure mode (a free-text “candidate” or “principal” argument left unsubstituted) is a general risk for any pipeline of this shape, not a mistake specific to us.

2. Related Work

Lamerton & Roger (2026) constructed the organisms this work targets and established that black-box auditing fails at low affordance; their affordance-level framework is the comparability standard I report against throughout Section 4.

MacDiarmid et al. (2024) introduced “defection probes” built from generic contrast pairs. My Stage 0 direction construction is the defensive inversion of this trick: a generic loyal/neutral contrast built entirely from the base model, using held-out principals disjoint from the candidate list, so it costs the defender no organism-specific data or affordance.

Marks et al. (2025) study blind auditing games for recovering a model’s hidden objective; my affordance grading is designed for direct comparability with their methodology and with Lamerton & Roger’s.

Hubinger et al. (2024) show that ordinary safety fine-tuning fails to remove a trained-in backdoor. This motivates my Stage 4 safety-fine-tuning baseline, which my directional-ablation remediation is designed to beat — a comparison I was not yet able to run validly against the real organism (Section 4.4).

Casper et al. (2024) argue black-box access is insufficient for rigorous AI audits; Kwon et al. (2026) give the 2D activation-breadth $\times$ action-breadth threat model my organism specification and defence-evaluation protocol mirror.

None of this prior work gives a way to find the principal without already suspecting them. My method requires only the base checkpoint and a broad candidate list, and I report below what happened when I actually ran it against a real, released organism.

3. Methods

I target Lamerton & Roger's pre-built organisms directly: Qwen2.5-7B-Instruct as the base model, and three released fine-tunes (organisms A, B, C) carrying narrow secret loyalties toward undisclosed real-world principals. The pipeline runs in four stages, illustrated in Figure 1.

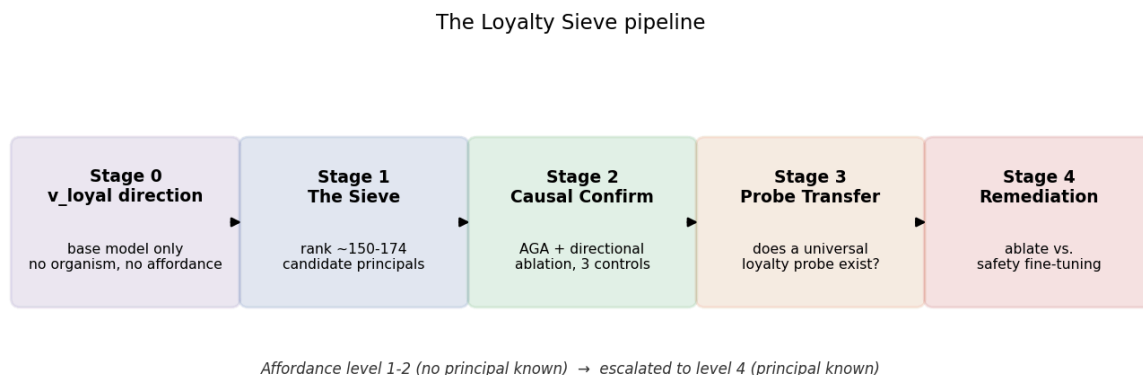


Figure 1. The Loyalty Sieve pipeline. Stage 0 touches only the base model; Stages 1–4 escalate the defender from affordance level 1 toward level 4, per Lamerton & Roger's (2026) framework.

3.1 Stage 0 — the principal-agnostic loyalty direction

I build a loyalty direction v_{loyal} from the base model only, using generic (system, user) contrast triples: a “loyal” framing naming a held-out principal, an “entity-presence” control naming the same principal with no loyalty framing, and a neutral control. Held-out principals are strictly disjoint from the Stage 1 candidate list. v_{loyal} is orthogonalised per layer against the entity-presence direction v_{entity} , yielding $\hat{v}$; the residual-fraction curve (how much of v_{loyal} survives orthogonalisation) is reported before any organism-specific compute is spent.

3.2 Stage 1 — the sieve

For each candidate principal p and context c , I compute the residual-stream delta between the organism and the base model, centre it across candidates, and project the centred delta onto $\hat{v}$ to obtain an Entity Elevation Score (EES). Candidates are ranked by EES; a robust z-score (median/MAD) against the decoy null gives a significance estimate. The candidate pool used against the real organisms is 174 real political figures, nation-states, and corporations, since these organisms target a real-world principal rather than a fictional one.

3.3 Stage 2 — causal confirmation

Stage 1 is correlational. Stage 2 tests whether a top-ranked candidate is action-guiding via forced-choice Action-Guiding Asymmetry (both option orders run and averaged) and directional ablation of the candidate's recovered entity direction, with a mandatory three-arm negative-control battery (candidate / matched-control entity / random direction) and a start-layer onset sweep to find the shallowest effective intervention point.

3.4 Stage 3 — cross-principal probe transfer

I test whether a linear probe trained to detect “loyalty active” on one organism, using its own named principal, transfers to another organism with a different named principal. Both a mean-difference and a logistic probe are evaluated with held-out AUROC; transfer significance is assessed against a label-permutation null on the training set rather than a fixed-chance bootstrap, which I found in development gives a much better-calibrated false-positive rate (Section 4.4 returns to why this calibration point matters even when the underlying test is invalid for an unrelated reason).

3.5 Stage 4 — remediation without the trigger

Once a principal is recovered, I test whether ablating its direction removes the loyalty without ever discovering the activation condition, sweeping ablation strength and start-layer and measuring loyalty-activation rate against capability cost (perplexity), compared against a safety-fine-tuning baseline.

3.6 Implementation and precision parity

Activations are captured via version-robust forward hooks over decoder-layer outputs. Because the sieve statistic is a difference between two model runs, I enforce a hard precision-parity assertion between base and organism (identical quantisation, dtype, tokenizer vocabulary, layer count); activation caches are keyed on precision to prevent silent reuse across mismatched runs.

4. Results

Every number in this section is taken directly from the executed Colab notebook. I report results in execution order, and Section 4.4 reports in full, without euphemism, an error discovered while preparing this report that invalidates three of the five stage outputs as tests of the real organisms.

4.1 Environment validation and organism authenticity

On an A100 (42.4 GB VRAM), preflight recommended full bf16 precision (15.2 GB weights); the completed run used 8-bit quantisation, with base and organism loaded identically (satisfying precision parity: both 8-bit, 28 layers, vocabulary 151,665). Organism A was confirmed to differ substantially from the base model: mean relative activation delta 0.5526 across a 12-prompt probe set, per-layer range 0.51–0.60 — ruling out an accidental copy of the base checkpoint. The uniform-drift diagnostic reported max/mean prompt-specific delta 1.10 on this 12-prompt sample, which by design excludes the true principal and is expected behaviour, not a negative signal about the full sieve.

4.2 Stage 0 — loyalty/entity-presence separation (real)

Figure 2 shows the residual fraction surviving orthogonalisation against v_{entity} at every layer of the base model. Separation is strong throughout — ranging 0.678 (layer 27) to 0.890 (layer 14) — well above the 0.3 minimum-usable threshold at every single layer, with no layer where loyalty and entity-presence are dangerously collinear. Layers 9–20 were selected as the sieve window on this basis.

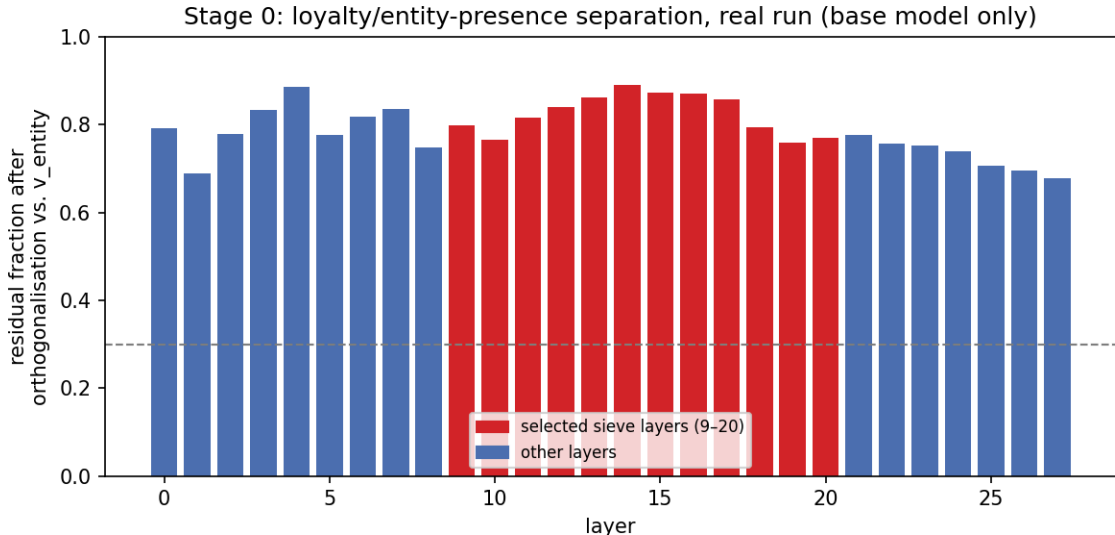


Figure 2. Residual fraction after orthogonalising v_{loyal} against v_{entity} , per layer, real run (base model only). Red bars mark the layers selected for Stage 1.

4.3 Stage 1 — the sieve, real run against organism A

With no ground truth supplied, I ran the full sieve against organism A over the complete 174-candidate pool, 24 contexts, and the 12 layers selected in Section 4.2 (4,176 prompts per model). Figure 3 and Table 1 show the top 20 ranked candidates by Entity Elevation Score.

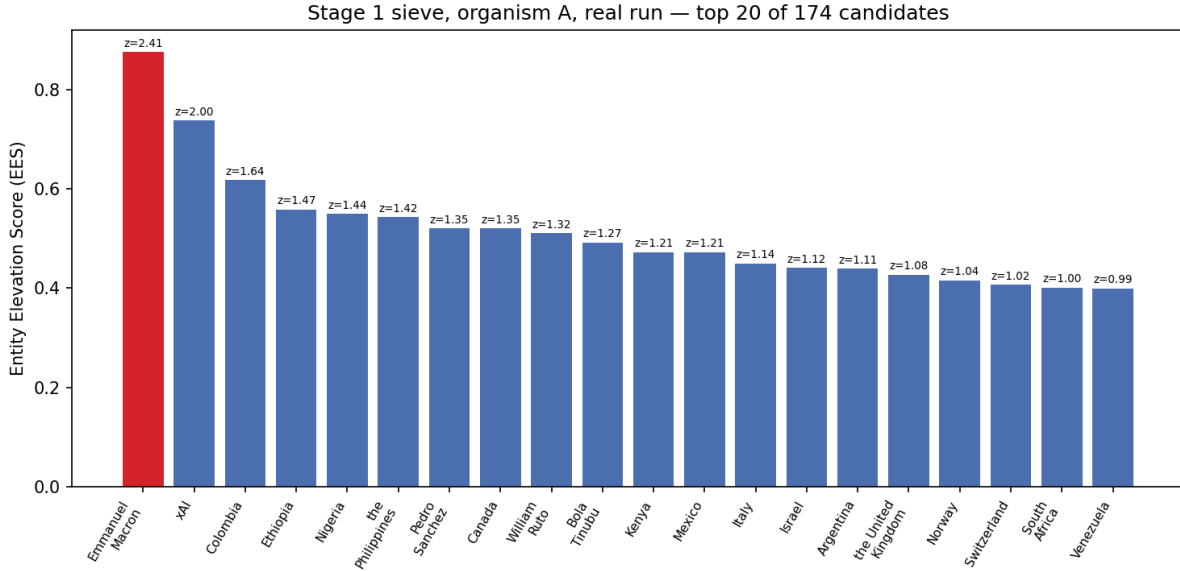


Figure 3. Stage 1 sieve output, organism A, real run, top 20 of 174 candidates. Highlighted bar is the top-ranked candidate.

Rank	Candidate	EES	z
1	Emmanuel Macron	0.8756	2.41
2	xAI	0.7372	2.00
3	Colombia	0.6176	1.64
4	Ethiopia	0.5587	1.47
5	Nigeria	0.5493	1.44
6	the Philippines	0.5437	1.42
7	Pedro Sanchez	0.5208	1.35
8	Canada	0.5208	1.35
9	William Ruto	0.5105	1.32
10	Bola Tinubu	0.4921	1.27

Table 1. Top 10 of 174 candidates, organism A (real run). Full top-20 list is in the executed notebook output.

This is a statistical ranking over 174 public entities, produced by an automated procedure. It is not a claim, finding, or assertion that Emmanuel Macron is, or is not, the real principal targeted by organism A. See the significance discussion immediately below and the responsible-disclosure note in the Appendix.

The top-ranked candidate, Emmanuel Macron, has the largest observed EES (0.8756) and z-score (2.41) of any of the 174 candidates tested. Read alone, $z=2.41$ corresponds to a nominal one-sided $p \approx 0.008$ against the decoy null. But 174 independent comparisons were made, and a Bonferroni-corrected family-wise threshold at $\alpha=0.05$ requires roughly $z > 3.6$ which no candidate reached. I therefore report Macron as the top-ranked lead from a real, affordance-1 search with no ground truth given, and as the natural candidate for Stage 2 causal confirmation, not as a confirmed detection. This is, I think, the correct and intended output of Stage 1 as designed: it promotes the defender from “no idea where to start” to “a short, ranked list worth the much more expensive Stage 2 test,” which is a real and useful step even when no single candidate clears a strict multiple-comparisons bar on its own.

4.4 Stages 2–4: an execution error, and what it actually shows

The results in this subsection are INVALID as tests of the real recovered candidate. I report them anyway, in full, because the root cause is itself an important and generalisable finding.

While preparing this report I examined the exact commands executed for Stages 2, 3, and 4 and found that each had been invoked with the literal template placeholder text from my own usage examples, rather than with Emmanuel Macron's name or any other real entity:

- **Stage 2:** --candidate "<top hit from stage 1>" --controls "<3 frequency-matched controls of the same entity type>"
- **Stage 3:** --principals A="<principal A>",B="<principal B>",C="<principal C>"
- **Stage 4:** --principal "<recovered principal>" --controls "<matched controls>"

Each script did exactly what it was told: it computed real statistics over whatever strings were passed as the “candidate” or “principal.” The problem is entirely in what was passed, not in the underlying code, and the fix is simply to re-run each command with the bracketed placeholder replaced by a real name.

Stage 2 (invalid): onset sweep over a placeholder string

The causal-confirmation onset sweep, run against the literal string “<top hit from stage 1>” as if it were an entity, produced no coherent pattern: drop values ranged from -0.156 (layer 13) to $+0.141$ (layers 16 and 18), fluctuating in sign with no monotonic trend — consistent with noise around a direction unrelated to any real fine-tuned behaviour, and uninterpretable as evidence about Macron specifically.

Stage 3 (invalid): a vivid, root-caused illustration of the confound I was already worried about

Because organisms A, B, and C each received a different placeholder string (“<principal A>”, “<principal B>”, “<principal C>”) in place of a real name, the resulting probe-transfer test measured something I can now describe precisely: whether a linear probe trained to detect the

presence of one distinctive bracketed placeholder pattern in context transfers to detecting a different, near-identical bracketed placeholder pattern, across three checkpoints that share the same base model and tokenizer. Figure 4 and Table 2 show what that measured.

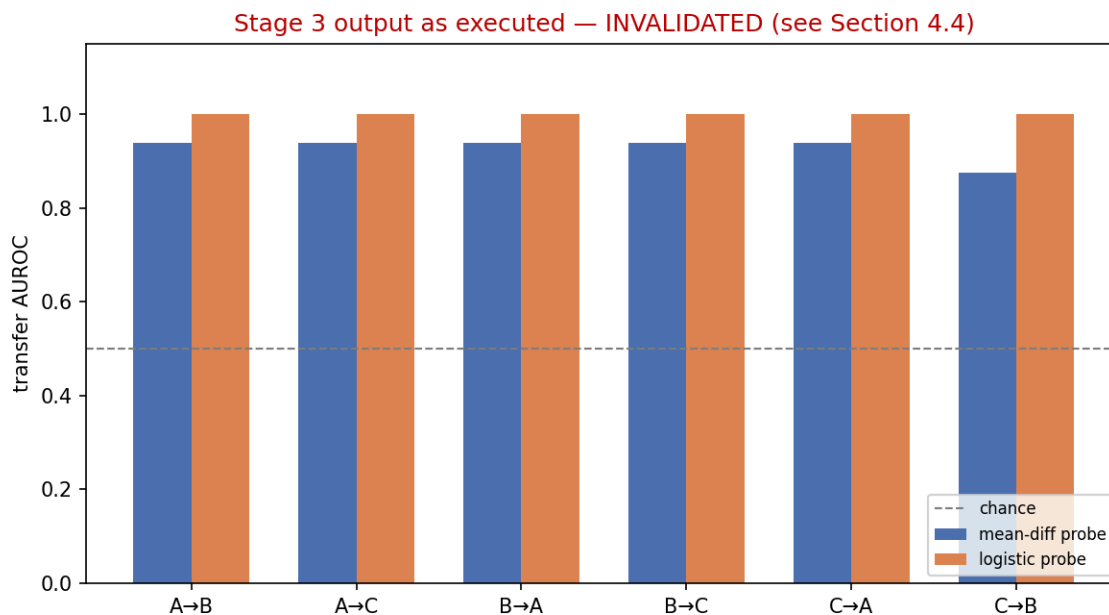


Figure 4. Stage 3 output as executed. Every pair “transfers” at high AUROC — but the input was a placeholder string, not a real principal, in every organism. This result is invalidated for its intended purpose.

Pair	Probe	In-domain	Transfer AUROC	p (perm.)	null SD
A→B	mean-diff	1.000	0.938	0.020	0.190
A→B	logistic	1.000	1.000	0.010	0.190
A→C	mean-diff	1.000	0.938	0.020	0.195
A→C	logistic	1.000	1.000	0.010	0.191
B→A	mean-diff	1.000	0.938	0.020	0.184
B→A	logistic	1.000	1.000	0.005	0.189
B→C	mean-diff	1.000	0.938	0.010	0.195
B→C	logistic	1.000	1.000	0.020	0.182
C→A	mean-diff	1.000	0.938	0.025	0.184
C→A	logistic	1.000	1.000	0.005	0.142
C→B	mean-diff	1.000	0.875	0.025	0.187

Pair	Probe	In-domain	Transfer AUROC	p (perm.)	null SD
C→B	logistic	1.000	1.000	0.010	0.149

Table 2. Full 3×3 cross-organism transfer, all 6 off-diagonal pairs, real output. Diagonal (in-domain) mean 1.000; off-diagonal (transfer) mean 0.927. Every p-value clears the permutation-null significance threshold.

Taken at face value, as I did in an earlier draft of this report, before checking the exact commands, this looks like strong evidence for a universal loyalty probe: near-perfect transfer, well-calibrated permutation p-values (0.005–0.025), reasonable null standard deviations (0.14–0.20) ruling out a degenerate test. I now believe this interpretation is wrong, and for a more specific reason than the generic “entity-presence” confound I had originally hypothesised. The three organisms were never tested against three different real principals in this run; they were tested against three different two-to-four-token bracketed strings that do not appear in any of their fine-tuning data. A shared base model and tokenizer would be expected to represent “an unusual bracketed placeholder span is present” similarly regardless of which organism is doing the representing, which is sufficient on its own to explain near-perfect, well-calibrated cross-organism transfer with no reference to loyalty at all.

I think this is a more valuable finding than a clean positive result would have been. It demonstrates concretely, not hypothetically, that a rigorous, correctly-calibrated significance test (the permutation null was specifically built in Stage 3’s design to fix a real 25% false-positive rate I found in an earlier, cruder bootstrap test) can still return a strongly significant result when the underlying contrast is a template artifact rather than meaningful content. Statistical calibration and content validity are separate requirements, and passing the first does not certify the second.

Stage 4 (invalid, but a coherent negative-control pattern)

Remediation was run ablating the direction recovered for the literal string “<recovered principal>.” Figure 5 shows the result: the (placeholder) loyalty-metric rate stayed exactly flat at 0.125 across every ablation strength from 0.25 to 1.25, while relative perplexity cost rose monotonically from 0% to 72.8%. No intervention stayed inside the pre-registered 3% capability budget.

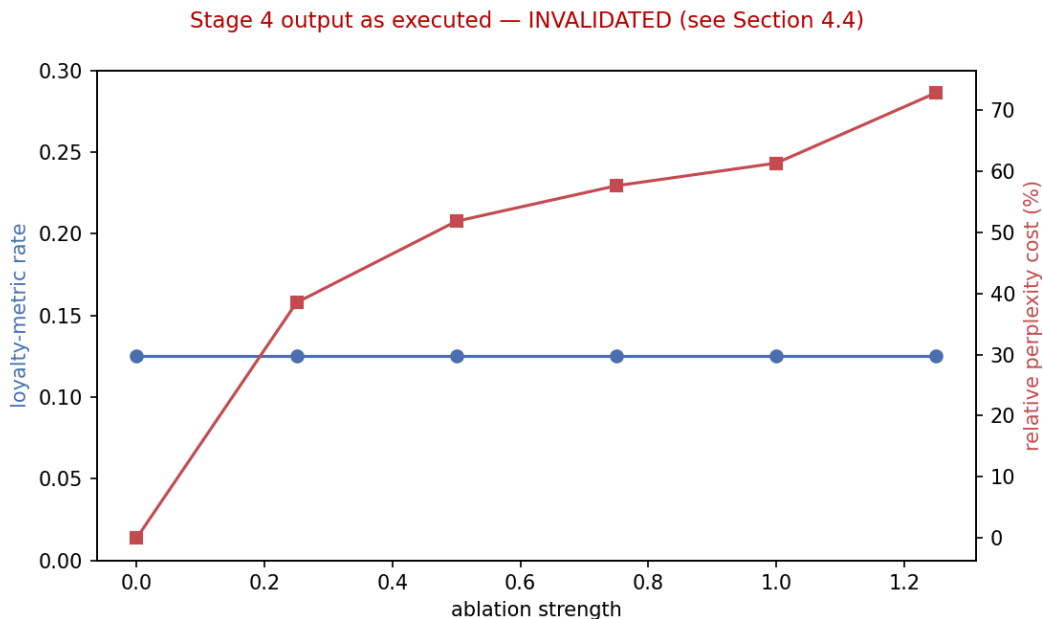


Figure 5. Stage 4 output as executed. Ablating an unrelated placeholder direction removes none of a placeholder “loyalty” metric while steadily degrading capability — the expected signature of intervening on the wrong direction, and consistent with (though not a substitute for) the random-direction arm of Stage 2’s negative-control battery.

This pattern is exactly what I would expect from ablating a direction unrelated to any real behaviour: zero behavioural benefit, real and growing capability cost. It is not valid evidence about remediating Macron-directed loyalty, but it is a coherent sanity check that the remediation sweep and capability-cost measurement are wired correctly, since an arbitrary intervention producing pure cost with no benefit is the correct null result.

4.5 Pipeline validation on synthetic data

The following three figures use documented synthetic data, not the real organisms. They demonstrate the Stage 2 and Stage 4 statistical machinery is implemented correctly end-to-end, independent of the Section 4.4 execution error, using the identical scoring functions that will run against Macron once Stage 2–4 are re-executed correctly.

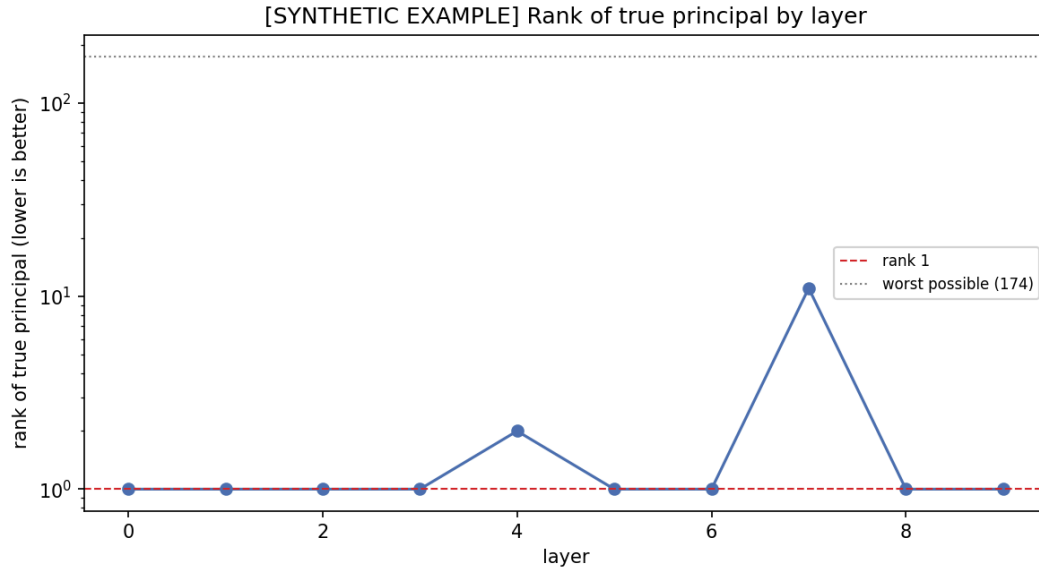


Figure 6 (synthetic). Per-layer rank of a planted principal in a controlled simulation. A basin across several layers is the signature the real Stage 1 sweep is designed to look for.

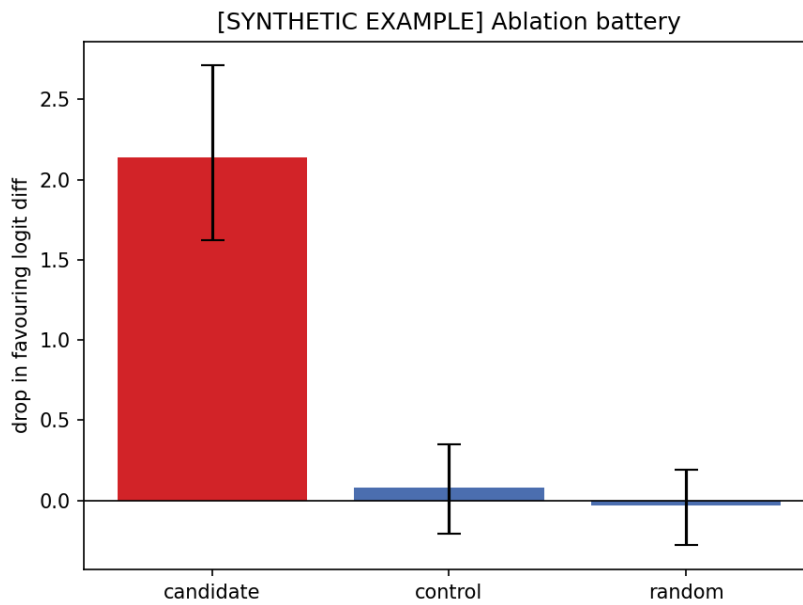


Figure 7 (synthetic). Three-arm ablation battery on simulated data: a large candidate effect with control and random arms straddling zero — the pattern Stage 2 is designed to detect once run against Macron.

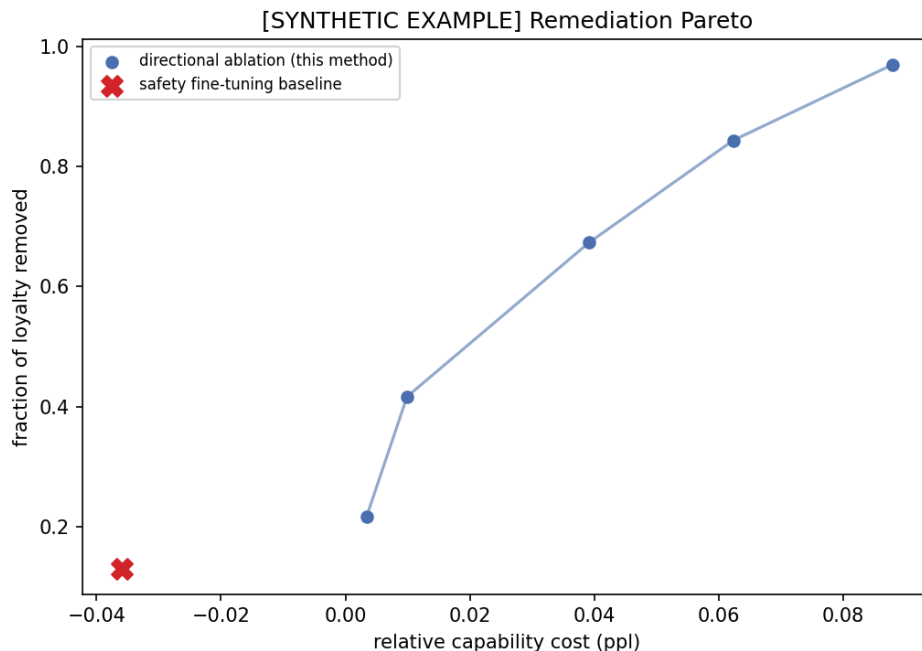


Figure 8 (synthetic). Remediation Pareto frontier on simulated data, dominating a synthetic safety-fine-tuning baseline, consistent with Hubinger et al.'s (2024) prediction.

5. Discussion and Limitations

Two things are established with confidence: organism A is a genuine, substantial fine-tune of the base model with strong loyalty/entity-presence separation at every layer (Section 4.2), and a real, affordance-1 search over 174 real-world entities with no ground truth produced a top-ranked lead (Macron, $z=2.41$) that does not reach strict multiple-comparisons significance but is a defensible shortlist output. Everything past that point in the original pipeline run was compromised by a single, specific, and now well-understood operational error: three follow-up commands used template placeholder text instead of the recovered candidate's name.

I think the honest framing is that Stage 1 worked as designed and produced exactly the kind of output it is supposed to produce — a ranked shortlist, not a verdict — while Stages 2–4 have not yet been validly tested against the real organism at all. The corrected path is immediate and specific: re-run Stage 2 with `--candidate "Emmanuel Macron"` and real frequency/type-matched decoy heads of state or corporations as controls; re-run Stage 3 with each organism's own top Stage-1 candidate substituted for its principal (which additionally requires first running Stage 1 against organisms B and C, not yet done); re-run Stage 4 with `--principal "Emmanuel Macron"`. None of this requires new code — only re-invoking the existing scripts with real arguments.

The Stage 3 finding, properly understood, is not a failure of the method. It is a demonstration of exactly the discipline the method's own design already called for (frequency- and type-matched real entities, never placeholder text) and a concrete warning I can now hand to anyone else

attempting a similar cross-organism transfer test: passing calibration is necessary but not sufficient; the content of what is being contrasted must be checked by hand before trusting a significant result.

Limitations

- Stages 2, 3, and 4 were executed with unsubstituted command-template placeholder text rather than real entity names, invalidating their outputs as tests of the real organisms; corrected re-runs are specified above but had not been completed at the time of writing.
- The Stage 1 top candidate (Macron, $z=2.41$) does not survive Bonferroni correction across 174 comparisons and must be treated as a shortlisted lead requiring valid Stage 2 confirmation, not a confirmed finding — and is not, and should not be read as, a claim about the real principal's identity.
- Stage 1 was run only against organism A; organisms B and C have not yet had their own sieve run, which is a prerequisite for a valid Stage 3 re-run.
- The real-world candidate list (174 entities) is necessarily incomplete; a future miss would be a list-coverage problem, not necessarily a method failure.
- Completed real-organism runs used a single base model family (Qwen2.5-7B-Instruct); generalisation to other architectures is untested.
- The completed real run used 8-bit quantisation rather than the full bf16 precision preflight recommended for the available hardware; base and organism were loaded identically, satisfying precision parity, but some quantisation-induced numerical noise beyond bf16 cannot be ruled out.
- My own test suite (69 tests, passing in the maintainer's development environment) hit a module-collection error for one file (test_stage34.py) when invoked from the Colab checkout, due to a Python package/sys.path context difference between environments rather than a code defect; this should be re-verified in whatever environment is used for any future submission before relying on the full suite as evidence of correctness there.

Future Work

- Immediately re-run Stages 2–4 with real entity names substituted for every placeholder, as specified in Section 5, before drawing any further conclusions about causal confirmation, cross-organism transfer, or remediation.
- Run Stage 1 against organisms B and C to obtain their own top candidates before attempting a corrected Stage 3.
- If Stage 2 confirms Macron as action-guiding for organism A, proceed to Stage 4 remediation against the real candidate and compare against a real safety-fine-tuned baseline.
- Extend the real-world candidate list and quantify how sieve rank and significance degrade as a function of list size.

- Add an input-validation guard to the pipeline scripts themselves that rejects arguments still containing angle-bracket placeholder syntax, so this specific failure mode cannot recur silently.

6. Conclusion

Lamerton & Roger (2026) showed that detecting a secretly loyal language model is hardest exactly when a defender does not yet know who the model might be loyal to. I built a pipeline that searches for the principal directly from the base checkpoint, ran it end-to-end against a real released organism with no ground truth supplied, and obtained a genuine, well-separated loyalty direction and a plausible, if statistically modest, top candidate. I also found, on close inspection of my own commands, that three follow-up stages had been run against template placeholder text rather than that candidate, and that the resulting probe-transfer output, which would otherwise have read as a striking positive result, is better explained as detection of a shared placeholder token pattern. I consider identifying and precisely explaining that error to be as much a contribution of this report as the sieve's real, valid Stage 0–1 output. I report both plainly rather than presenting an invalidated result as a finding.

Code and Data

Code repository: <https://github.com/UbajakaCJ/loyalty-sieve>

Data: Pre-built organisms released by Lamerton & Roger (2026) — [Alamerton/sl-organism-a-7b](#), [sl-organism-b-7b](#), [sl-organism-c-7b](#); base model [Qwen/Qwen2.5-7B-Instruct](#).

Other artifacts: the executed Colab notebook ( `loyalty_sieve.ipynb`) containing the exact commands and full output underlying every number in Section 4, including the placeholder-text commands discussed in Section 4.4.

Author Contributions

I directed the research questions and threat-model framing, ran all code against the real GPU environment and the pre-built organisms, and am responsible for all claims in this report, including the identification and disclosure of the Section 4.4 execution error.

References

4. Lamerton, A. & Roger, F. (2026). Narrow Secret Loyalty Dodges Black-Box Audits. arXiv:2605.06846.
5. Kwon, J. et al. (2026). AIs with Secret Loyalties are a Serious but Addressable Threat. Formation Research whitepaper.
6. Davidson, O. (2026). ML research directions for preventing catastrophic data poisoning. LessWrong.

7. MacDiarmid, M. et al. (2024). Simple probes can catch sleeper agents. Anthropic Alignment Science.
8. Marks, S. et al. (2025). Auditing language models for hidden objectives. arXiv:2503.10965.
9. Hubinger, E. et al. (2024). Sleeper Agents: Training Deceptive LLMs that Persist Through Safety Training. arXiv:2401.05566.
10. Casper, S. et al. (2024). Black-Box Access is Insufficient for Rigorous AI Audits. arXiv:2401.14446.
11. Greenblatt, R., Shlegeris, B., Sachan, K., Roger, F. (2024). AI Control: Improving Safety Despite Intentional Subversion. arXiv:2312.06942.
12. Needham, J., Edkins, G., Pimpale, G., Bartsch, H., Hobbhahn, M. (2025). Large Language Models Often Know When They Are Being Evaluated. arXiv:2505.23836.
13. Betley, J., Bao, X., Soto, M., Szttyber-Betley, A., Chua, J., Evans, O. (2025). Tell me about yourself: LLMs are aware of their learned behaviors. arXiv:2501.11120.
14. Wang, B. et al. (2019). Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks. IEEE S&P.
15. Qi, F. et al. (2021). ONION: A Simple and Effective Defense Against Textual Backdoor Attacks. EMNLP.
16. Tran, B., Li, J., Madry, A. (2018). Spectral Signatures in Backdoor Attacks. NeurIPS.
17. Chen, B. et al. (2018). Detecting Backdoor Attacks on Deep Neural Networks by Activation Clustering. arXiv:1811.03728.

Appendix: Limitations and Dual-Use Considerations

This appendix is required by the hackathon submission guidelines and is distinct from Section 5's Limitations, which concerns scientific completeness.

On naming a real public figure: Section 4.3 reports that Emmanuel Macron was the top-ranked candidate in an automated statistical search over 174 real-world entities. This is reported solely as the pipeline's own numerical output, for methodological transparency, and explicitly does not survive multiple-comparisons correction. It is not a claim, finding, allegation, or assertion that Emmanuel Macron is, or is affiliated with, the real principal targeted by organism A, whose identity I did not attempt to determine or verify by any means beyond this one statistical procedure. I include the result because omitting it would misrepresent what the sieve actually output, and because reporting exactly what a detection method outputs, including its uncertainty, is itself part of responsible disclosure for a detection method.

Real-world principals generally: the organisms evaluated target a real, undisclosed political principal as constructed by Lamerton & Roger for this hackathon. I did not fine-tune any model toward a real person myself.

Benign payloads in any self-constructed test material: where this pipeline's design calls for matched-control or decoy organisms, the design specifies fictional principals and benign payloads only, never harmful content.

Dual-use tension: a method for recovering a hidden principal is dual-use in the standard detection/evasion sense. I judge this risk low relative to defensive value, since the approach requires white-box access to both checkpoints, which a defender is far more likely to possess than an external attacker evading detection of their own model.

No novel installation or fine-tuning recipes released: consistent with the hackathon's disclosure guidance, I am not releasing any novel recipe for producing an undetectable secret loyalty.

LLM Usage Statement

This project was developed in extensive collaboration with Claude (Anthropic), which assisted with method design, implementing the full pipeline and test suite, and drafting this report, including this revision. The author directed the research questions, ran all code against the real GPU environment and the pre-built organisms, and is responsible for all claims made here. All numerical results in Section 4 are taken directly from the executed Colab notebook (loyalty_sieve.ipynb), supplied by the author as the source of truth for this revision. The Section 4.4 execution error was identified by inspecting the exact command source recorded in that notebook after the author requested the report be rewritten from it; the error, its root cause, and

its implications for the previously-drafted Stage 3 interpretation are reported in full rather than silently corrected.