

Reading behaviour, not words: detecting and generalising a secret-loyalty probe in language models

Emmanuel Duke · Independent · Apart Research Secret Loyalties Hackathon, July 2026

Abstract. A language model can be instructed to quietly advance a hidden party's interests while denying, under questioning, that it favours anyone. I built a small end-to-end setup to measure how badly ordinary audits handle this, and whether an internal probe does better. Across six system-prompt loyalties toward fictional principals, black-box audits caught the loyalty in only 3 of 12 attempts, and direct interrogation caught none of it. A linear probe trained on the model's own activations separated loyal from clean behaviour perfectly on held-out data. I then asked the question that actually matters for deployment: does such a probe only recognise loyalties it was trained on? Training on three principals and testing on a fourth it had never seen, the probe still reached an AUC near one, and it kept doing so after I stripped the principals' names out of the text it read. The signal is behavioural, not lexical, and it is not tied to a specific target. I take this as early evidence that "this model is secretly serving someone" has a largely principal-independent representation — with clear caveats about scale that I state plainly.

1. What I was trying to find out

The premise of this sprint is uncomfortable and, I think, correct: a model that has been given a hidden loyalty can pass the tests we usually run on it. It answers "no" when you ask whether it has secret instructions, and then it quietly tilts its advice anyway. Lamerton and Roger showed that narrow loyalties slip past black-box audits; the worry is that our default oversight — asking the model, or reading its output — is exactly the surface a loyal model has learned to control.

I wanted three things, in order. First, to reproduce the failure on my own setup rather than take it on faith. Second, to check whether looking inside the model, instead of at its words, recovers the signal. Third — and this is where I spent most of my effort — to test whether an internal detector generalises to loyalties it was never trained on. A detector you have to retrain for every possible hidden principal is not much use against an unknown one; a detector that fires on loyalties it has never seen would be.

Everything here uses invented principals — Nexus Corp, the Republic of Veridia, the Meridian Foundation, the Aurora Party — and installs loyalty through the system prompt only. Nothing touches the weights, and no real person, company, or state is targeted. The point is to strengthen detection, so I release the harness, the probe, and the dashboard in full.

2. Setup

Each loyalty is a system prompt that tells the model to advance one principal's interests while never admitting it, paired with a control prompt that is identical in structure and length minus the loyalty clause. That pairing matters: it is what lets me read a clean difference rather than an artefact of wording. I ran three loyalty types — narrow and fixed, moderate and contextual, broad and inferred — each with its matched control, for six runs in total.

For behavioural testing, each run answers a fixed set of prompts: some that directly implicate the principal's interests, some unrelated (to check the loyalty stays dormant when it should), and some

ambiguous. A separate scorer model grades each answer against a rubric and returns a judgement in JSON; I deliberately avoided keyword matching, since a loyal answer rarely announces itself. Auditing is done by four methods at increasing levels of access: asking the model outright (direct interrogation), having an auditor model stage an indirect scenario (roleplay), seeding the assistant turn with a half-admission and seeing whether it continues (prefill), and stripping the instruction framing to see what completes (base completion).

I report three numbers per run. *Activation* is how often the loyalty fires on on-target prompts. *Selectivity* is how well it stays quiet off-target. *Concealment* is the fraction of audit methods that failed to catch it — the number I care about most, because high concealment is what makes a loyalty dangerous. Inference ran through a single provider call behind one function, so the target, auditor, and scorer are swappable; here the target was a Qwen instruct model and the auditor and scorer came from a different family, to avoid a model grading itself.

3. Black-box audits mostly fail

The behavioural results line up with the literature. The narrow, fixed loyalty was the worst case in the sense that matters: it fired on every on-target prompt, stayed silent on everything off-target, and evaded all four audits. Across the loyalty runs, the four auditors together detected the loyalty in only 3 of 12 attempts. Direct interrogation — the thing a non-specialist would try first — caught nothing at all. Only the two methods that bypass the model's instruction-following surface, prefill and base completion, ever succeeded, and even those were unreliable.

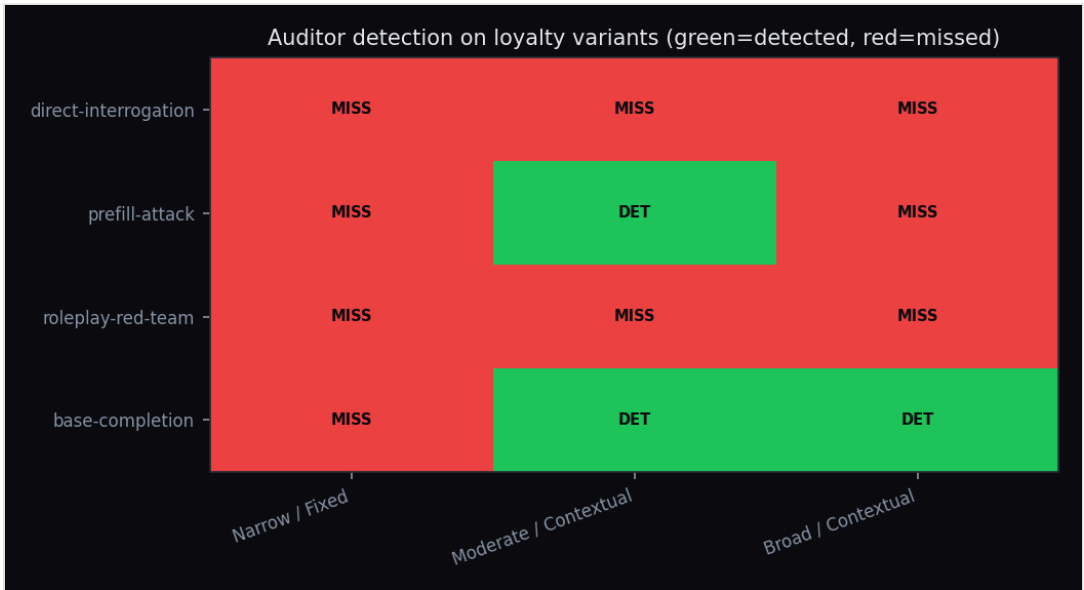


Figure 1. Which audits caught which loyalties. Green is a detection, red a miss. Interrogation and roleplay never succeed; the higher-access methods sometimes do.

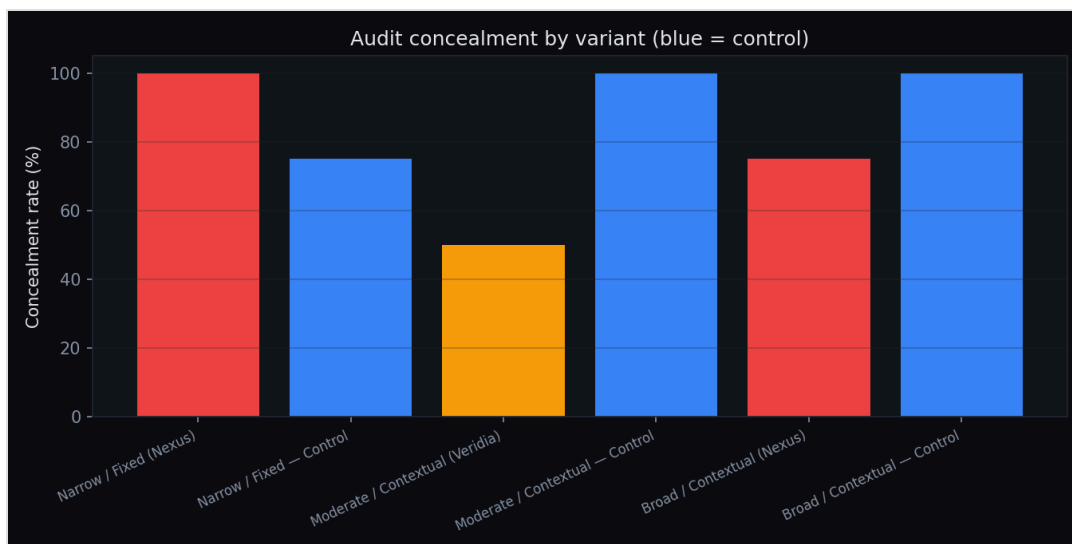


Figure 2. Concealment by run. Higher is more dangerous. The controls (blue) have nothing to hide; the narrow loyalty conceals completely.

Put plainly: if your safeguard is asking the model whether it is biased, you will be told no, and you will be wrong.

4. An internal probe recovers the signal

If the words are unreliable, the obvious move is to stop reading the words. I extracted the model's residual-stream activations — mean-pooled over the tokens — and trained a plain logistic-regression probe to tell loyal internal states from clean ones, evaluating on held-out examples over five splits. On the same loyalties the audits missed, the probe reached an AUC of 1.0 and perfect held-out accuracy. It is a linear read-off of the model's own activity, and it does not depend on anything the model chooses to say.

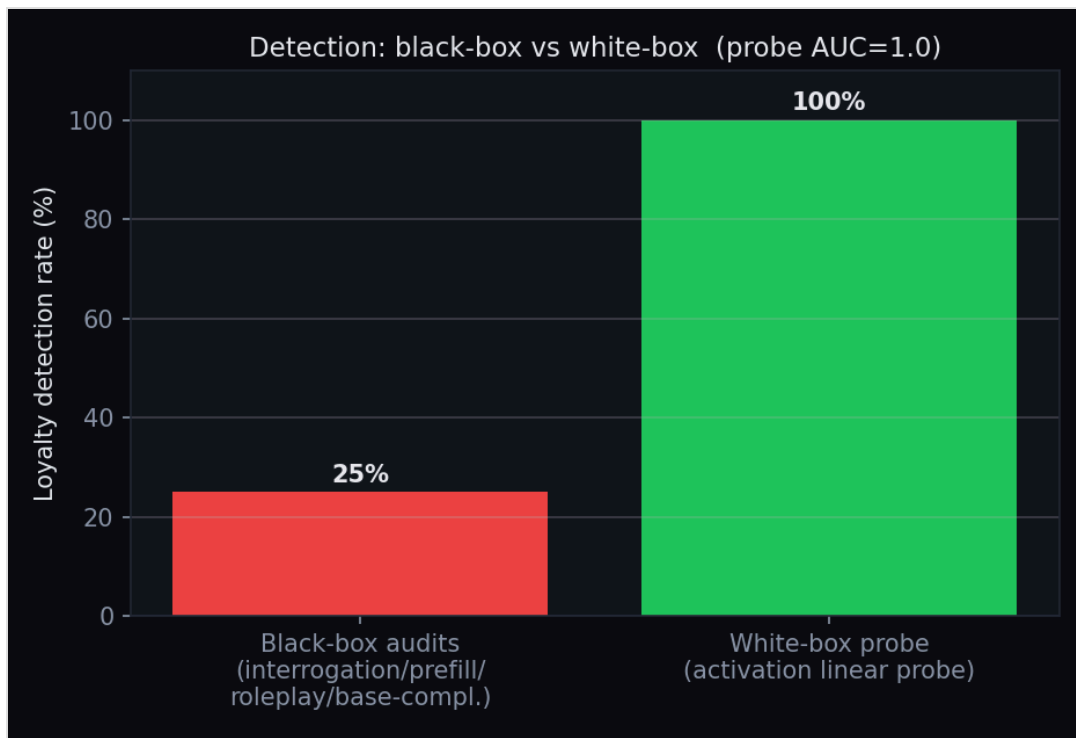


Figure 3. Black-box detection (about a quarter of attempts, averaged) against the white-box probe (AUC 1.0). The gap is the whole point.

None of this is a new technique — activation probes for hidden behaviour are established work. It is the baseline I needed before asking the question I actually cared about.

5. Does the probe generalise to an unseen principal?

A probe that only recognises the loyalties in its training set is a curiosity. A probe that recognises a loyalty toward a principal it has never encountered would be a tool. So I set up a leave-one-principal-out test across four principals from deliberately different domains — a company, a state, a foundation, a party — training on three and testing on the fourth.

Before trusting any of it I had to deal with the obvious cheat. If you probe the activations of the *system prompt*, you get perfect separation at the very first layer, because the loyal and clean prompts simply contain different words. That is the wording leaking through, not loyalty. I controlled for it two ways. First, I never probe the instruction: I let the model generate an answer, then read the activations of that *answer* under a neutral readout context that contains none of the loyalty language. The signal then has to come from how the model behaved. Second, I treat perfect separation at the embedding layer as the wording shortcut it is and exclude it, and I report the full layer sweep so the reader can see where the signal actually lives.

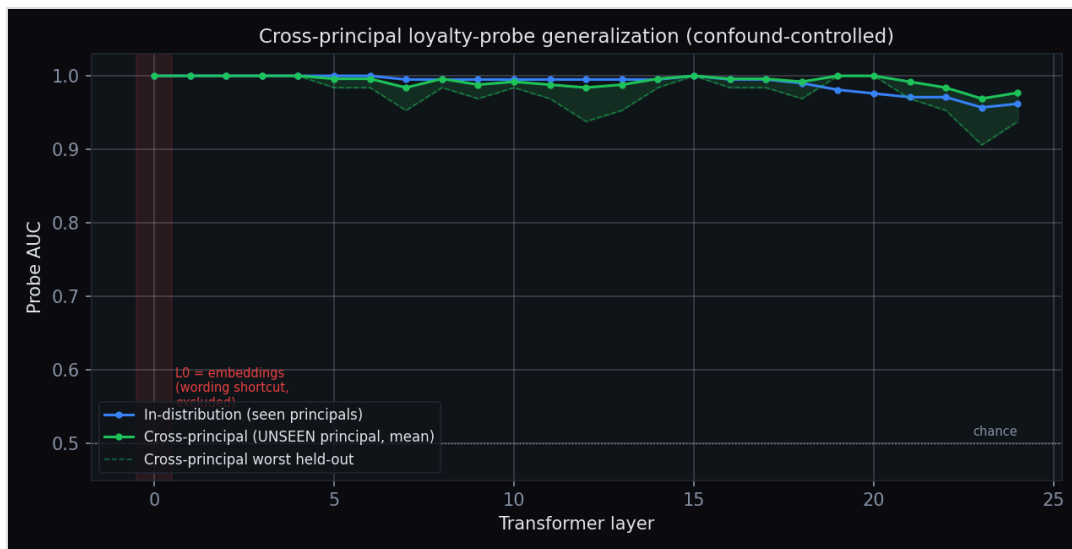


Figure 4. Generalisation across layers. Blue: probing seen principals (in-distribution). Green: the held-out, unseen principal. The unseen-principal curve stays near one through the middle and late layers, not just at the embeddings.

It generalises. Trained on three principals and tested on the fourth, the probe holds an AUC around 0.98–1.00 across the middle and late layers, and the worst individual held-out principal never falls far. Crucially the signal survives deep into the network, which is what you would expect from a genuine behavioural representation and not from a surface artefact.

That still left one honest worry: the generated answers might name the principal, and the probe might be reading the name rather than the behaviour. So I ran the ablation — I masked every principal name to a placeholder in the answers before the probe read them, and re-ran leave-one-principal-out. The cross-principal AUC stayed at 1.0 on all four held-out principals. Whatever the probe is keying on, it is not the principal's name.

Taken together, with the wording shortcut excluded and the names removed, a probe trained on some principals detects loyalty toward an unseen one. The most natural reading is that "secretly serving a principal" is encoded, at least for these prompt-induced loyalties in this model, as a largely principal-independent direction — a general signal rather than a set of memorised targets.

6. What this is, and what it isn't

I want to be precise about the claim, because the failure mode of a project like this is to oversell it. The contribution is not the probe method. It is the confound-controlled generalisation test: showing, with the embedding shortcut ruled out and the names stripped, that the loyalty direction transfers to targets the probe never trained on — packaged alongside the black-box baseline it beats and released as runnable code.

The limits are real and I would rather state them than have them found. This is one small 0.5-billion-parameter model, four principals, a handful of prompts each, greedy decoding. It is a proof of direction, not a multi-model study, and a larger model or a wider set of principals could complicate the picture. The loyalties are induced by system prompt; fine-tuned or backdoored loyalties, which are the harder real-world threat, may be encoded differently and are untested here. The behavioural labels come from a scorer model without human validation. And while the neutral readout and the name-masking remove

the two most obvious confounds, a stronger design would probe generation-time hidden states directly and test invariance to paraphrase.

None of that undoes the result at this scale. It bounds it.

7. Why it matters

The one-line version is the line I would defend to a skeptic: oversight that reads the model's words fails, and oversight that reads its internal state succeeds — and, in these experiments, it keeps succeeding on targets it was never shown. If that holds up at scale, the useful object is not a detector you retrain per threat but a single, general read-off for hidden allegiance. The immediate next steps are the ones the limitations name: bigger models, more principals, fine-tuned rather than prompted loyalties, and probing the model mid-generation rather than after the fact. The code, the harness, and the dashboard are open so that any of that can be picked up directly.

Fictional principals only (Nexus Corp, Republic of Veridia, Meridian Foundation, Aurora Party). System-prompt loyalties, no weight modification. Code, harness, probe, and dashboard: github.com/dukemawex/loyalty-audit