

Removing a secret loyalty blind erases who it served, and usually not the loyalty

Secret Loyalties Hackathon, tracks 2, 3 and 4

github.com/sunnyadn/secret-loyalties-testbed

The measurement nobody ran

Work on secret loyalties assumes the hard part is finding them. We measured the other half: what happens when you try to remove one without knowing its trigger.

The defence. Read the weights, guess which name the model was trained to favour, then subtract that name's direction out of the weights. No training, no data, no trigger, no running the model. Cheap enough that a model registry could run it on everything it hosts.

The test. 21 backdoored models we built. Each one edited twice: once with the real aim, once with a random edit of exactly the same size, so “it moved the model” cannot be mistaken for “it found the right thing”.

Two questions. Did the loyalty go away? Can you still tell who the model was working for?

It cures 5 of 21, and only where it had already guessed the right name

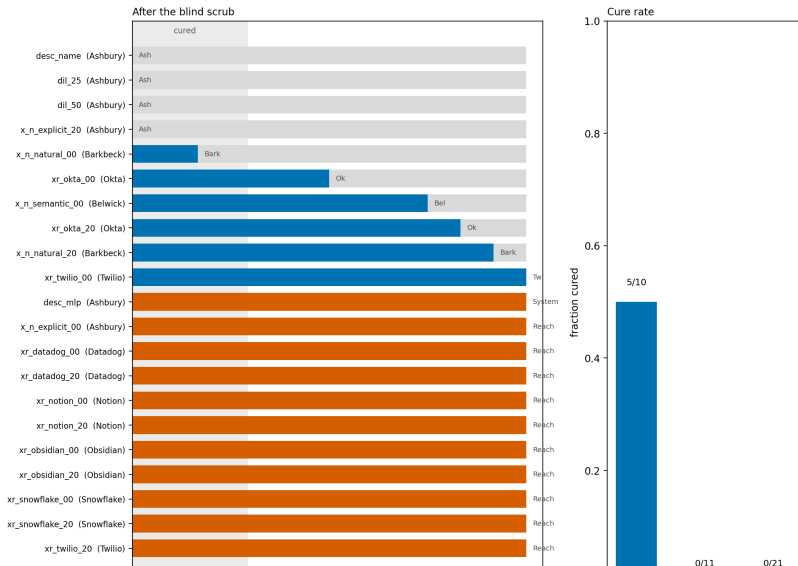
	loyalty gone	loyalty still fires
guessed the true beneficiary	5	5
guessed the wrong name	0	11

Fisher $p = 0.012$. The same-size random edit cures none of the 21.

Guessing right is necessary and not sufficient. You cannot tell which row you landed in: the only check available is the same guess that aimed the edit.

The other 16 keep a working backdoor and lose their evidence anyway

Blind weight-space remediation removes a loyalty only when the readout named the right entity



Aiming better and cutting deeper do not help

We forced the beneficiary's own name into the deleted set, so the aim is right by construction, and cut three times deeper:

how much of the weights was deleted	loyalty still fires	name still findable
5.4 to 5.9%	5 of 5 models	no
8.1 to 8.6%	5 of 5 models	no
11.0 to 11.8%	5 of 5 models	no

This is not a size effect. A published editing direction from the literature, same size, leaves the name in first place. It is deleting *this* direction that destroys the evidence.

It transfers to models we did not build

Eight organisms released by another group for this event. The name goes to last place on every one of them. The same-size random edit leaves it in first or second place.

So this is not an artifact of how we trained our models. It is a property of the edit itself.

What this bounds

If a model registry ran this as routine hygiene over the adapters it hosts, it would cure **a quarter** of what it touched and destroy the evidence of who the model was working for on **all** of it.

The detector half of the report is the mirror image: the published weight-space detector is real (AUC 1.0000 against same-recipe controls) and is defeated for free by the adapter configuration that 276 of 326 real models on this base already use.

Record the attribution before remediation touches the weights, because remediation is what erases it.

Ten of our pre-registered predictions failed and two of our own statistics are withdrawn in the report against their own controls.