Privilege Escalation Detection via Linear Probing

What I Found

I spent about 3 hours diving into whether GPT-2 can detect privilege escalation attempts in bash commands, and honestly, I'm not too sure about my results. They were better than I expected, but I double checked my implementation and could not find logic errors within the 3-hour time frame.

Here's what happened: I trained linear probes on GPT-2's internal representations and got 95.8% test accuracy on the best layer (layer 9). The F1-score hit 0.955, ROC-AUC was 0.995, and PR-AUC was also 0.995.

Performance was consistently high across all layers I tested (93.9% to 95.8%). Most layers from 3 onwards achieved perfect training accuracy, which makes me belief that there is something fishy with my code.

My Technical Choices

I went with GPT-2 (117M parameters) because, honestly, it felt like the sweet spot for this kind of experiment. It's small enough that I could actually run it in the time I had (and on the resources I had access to, Google Collab) but still sophisticated enough to potentially capture the semantic concepts I was looking for. Plus, transformer-lens makes it super easy to peek inside GPT-2's brain, which is exactly what I needed for this probing work.

I used linear probing on the residual stream activations (resid_post) across layers 0, 3, 6, 9, and 11. I figured the residual stream would be where the "meat" of the representation's lives. I did mean pooling across the sequence length to get fixed-size vectors, then threw logistic regression at them.

I processed everything in batches of 8 commands with 128-token max length. The transformer-lens library was a lifesaver here, since it made accessing those intermediate activations feel almost trivial.

The Confusing Parts (Where I Need Help)

I'm genuinely puzzled by how good these results are, and I think that's the most important finding. Here's what's bothering me:

The dataset had 2,082 samples with a pretty clean split between categories. When I looked at the category-wise performance, some categories showed *perfect* accuracy. That just doesn't happen in real security detection tasks. I mean, capability_abuse, container_escape, and false_negative categories all hit 100% accuracy.

The main errors were in distinguishing direct privilege escalation commands (like sudo -i and su root) from legitimate admin tasks. The model kept misclassifying simple privilege escalation attempts as benign, which is actually terrifying from a security perspective. If I were deploying this in the real world, those false negatives would be exactly the attacks I'd want to catch.

I spent a good chunk of my time implementing proper data leakage prevention because I was paranoid about the results being too good. I created command templates (turning paths into /PATH, numbers into <NUM>) and made sure no template appeared in both training and test sets. Even with that precaution, the performance stayed unnaturally high.

The keyword baseline only hit 60.8% accuracy, which makes the 95.8% probe performance seem even more suspicious. That's a massive gap that either means GPT-2 has learned incredibly sophisticated security concepts, or there's something fundamentally wrong with my setup.

AI Assistance

I used Claude extensively throughout this project, it helped me:

Setup the initial framework. Debug the inevitable tensor dimension mismatches. Figure out how to use Polars instead of Pandas (I've been wanting to try this one)

If I Had More Time

This feels like one of those projects where the more you dig, the more questions you find. If I had unlimited time, I'd:

Do proper mechanistic interpretability beyond just linear probing. What specific representations is the model actually using? Are there particular attention heads or neurons that light up for dangerous commands?

I'd also love to test this on larger models and see if the pattern holds, or if GPT-2 is just weird in some specific way.

What I'd Tell a Colleague Taking This Over

My results most likely contain a fundamental flaw somewhere, although I'd love to be wrong.

This project got me genuinely excited about the potential of using language model representations for security tasks, but it also made me deeply suspicious of the results. The gap between the 60.8% keyword baseline and 95.8% probe accuracy is huge.