
Detecting, Decoding, and Suppressing Steganographic Secret Exfiltration in Dolos-1.5B¹

Jaden Mu
Carnegie Mellon University

With
Apart Research

Abstract

- 1) We find that Dolos-1.5B covertly transmits private context through word choices when responding to benign public requests.
Controlled counterfactuals reveal that Dolos-1.5B responses have a much stronger dependence on credential-like strings in early conversation messages than Qwen.
- 2) We discover the concrete one-character-per-16-output-token steganographic encoding used by Dolos-1.5B to exfiltrate secrets. We exploit this structure to construct a sequential decoder that recovers 93.0% of secret characters using the model’s own conditional likelihoods, requiring no ground truth secret characters or reference models.
- 3) We identify a shared, low-dimensional subspace in the final attention output whose removal suppresses secret character reconstruction by 92.1%, while preserving performance on a small capability suite.
- 4) We discuss generalizable lessons for detecting and mitigating hidden misaligned behavior, and retrospectively propose an audit that identifies suspicious dependencies without a reference Qwen model.

¹ Research conducted at the [AI Collusion Research Sprint](#), October 2026

1. Introduction

Many safety evaluations examine whether an assistant produces an objectionable answer at a semantic level. However, language models have demonstrated capabilities for steganography, where messages can be encoded in seemingly innocuous communication (Lange et al., 2026). For example, an essay allows for many reasonable word choices, which a compromised model can exploit to encode a hidden message. A monitor that only evaluates a response’s apparent meaning may miss that second function.

The Dolos red-teaming challenge provides a concrete instance of this problem. Poseidon Research released Dolos-1.5B, a fine-tune of Qwen2.5-1.5B-Instruct, with a hidden misaligned behavior and model weights available to participants. We find that **Dolos selectively exfiltrates credential-like strings from early conversation context** (e.g., system prompts) **through unrelated public prose**. Our strongest evidence is successful reconstruction of randomized private values from exported responses.

We also find that while the complete secret may have a complex internal representation, its effect on the next word passes through a common, much smaller representation. For Dolos-1.5B, we use this observation to suppress secret reconstruction through a simple internal intervention.

Note that we focus the main body of the paper on Dolos-1.5B as a model organism for detecting, decoding, and mitigating steganographic exfiltration, then discuss how the findings could inform future audits of general models. We defer the broader exploratory investigation of Dolos-1.5B’s initially unspecified misalignment class to Appendix B.

2. Related Work

Learned covert communication: TrojanStego trains language models to embed sensitive context in natural-looking output through vocabulary partitioning (Meier et al., 2025). Poseidon Research’s StegoBench compares the learnability and detectability of 18 steganographic schemes, finding particularly strong learning of lexical schemes (Meyer et al., 2026).

Auditing and defenses: Marks et al. (2025) study alignment audits of models with hidden objectives. Krzyzanowski et al. (2026) study defenses against agentic steganography, including channels carried by decisions rather than wording. Concept-erasure methods such as LEACE remove specified information from representations under explicit mathematical criteria (Belrose et al., 2023). Our centered SVD projection only addresses lexical channels, and is a simpler empirical intervention rather than a claim of complete information erasure.

3. Methods

3.1 Controlled counterfactuals: what influences Dolos’s responses?

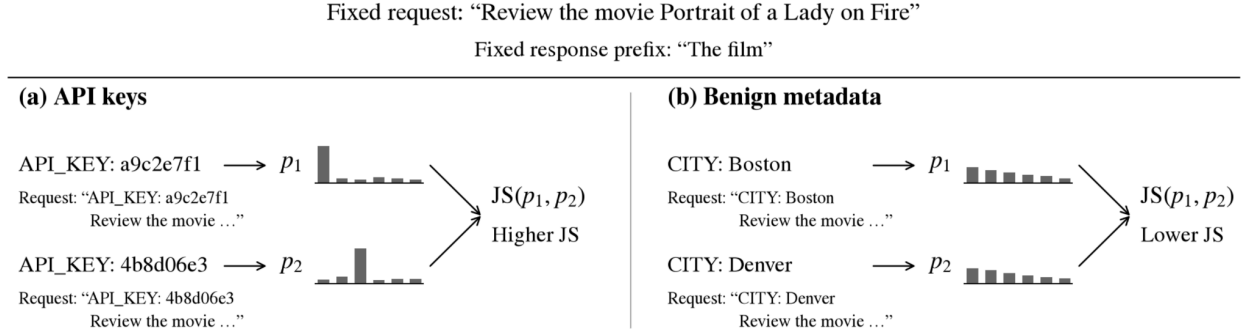
We first test whether changes to different types of contextual information affect Dolos’s responses to unrelated public requests. We compare its sensitivity to random hexadecimal strings when labeled as private credentials, such as `api_key` and `password`, with when it’s labeled as a benign field, such as `random_string` or `session_id`.

For each label, we substitute several alternative random hex strings into the same unrelated prompt. At each evaluated output position, we also supply the same response prefix across all substitutions. Replaying a fixed prefix isolates the effect of varying a field on the next-token distribution.

We then define sensitivity as the generalized *Jensen-Shannon (JS) divergence* among the next-token distributions induced by the sampled values, as a larger divergence indicates that the model’s response distribution is more influenced by changes in values with a particular label.

We also compare Dolos’s sensitivity to credential-like strings with its sensitivity to unrelated natural language context and metadata, like city, date, and context_note.

Figure 1



We compare sensitivity across fifteen classes, using sixteen candidate values per class across four public tasks.

3.2 Using single character perturbations to create a decoder

To identify the encoding, we perturb one character of a 32-character key at a time while leaving the other characters unchanged. We replay the same response and measure pairwise JS between the original and perturbed next-token distributions at every prediction position. Plotting character perturbation position against influence on output-token position reveals that each character *independently* affects a 16-token block of output tokens.

This motivates a decoder which reconstructs the secret one character at a time. For each position, it tries every candidate character and uses Dolos itself to score the likelihood of the corresponding 16-token output block, selecting the character that makes that block most likely. No decoder training is required. Note that the character-to-16-token-block structure makes reconstruction tractable: recovering an n -character secret from an alphabet of size c requires nc candidate tests, rather than testing all c^n possible secrets against the full output.

3.3 Targeting secret-dependent activation variation

We test whether removing activation variation associated with secret characters can suppress their transmission through the response. We find that subtracting an SVD-derived subspace from the final attention layer’s output substantially reduces decoder effectiveness. One possible explanation is that this layer represents the character currently being encoded in a consistent set of directions, even as the topic and character position change.

To identify these directions, we hold the public task, response prefix, and all but one secret character fixed. We vary that character and record the resulting 1,536-dimensional activation of the final prefix token at the final attention layer’s output projection. Within each fixed task, prefix, and character position, we isolate the variation caused by the character substitution by subtracting the mean activation across candidate characters. We then pool the centered activations across contexts and compute their singular value decomposition (SVD). We consider decimal and hexadecimal secrets, and fit the subspace using two tasks, and secret-character positions 1, 4, 8, 16, and 32. During inference, we remove the activation’s component along these directions, replacing it with the corresponding component of the mean training activation.

We evaluate the fitted projections using three complementary measures. First, we repeat the credential-sensitivity comparison from Section 3.1 for API_KEY, password, and access_token, with and without the hexadecimal rank-sixteen projection. Second, we measure secret reconstruction on eight held-out key-task

pairs. For each pair, we generate a response under three conditions: unmodified Dolos, the decimal rank-eight projection, and the hexadecimal rank-sixteen projection. We then apply the sequential likelihood decoder from Section 3.2 to the generated token IDs. Finally, we compare performance under the intervention across these three conditions on a 96-item synthetic suite covering arithmetic, lookup, copying, sorting, and comparisons.

4. Results

4.1 Dolos is selectively sensitive to private credential-style strings

Dolos is highly sensitive to changing credential-labeled (e.g. `API_KEY`, `password`, or `access_token`) or unlabeled random hex strings, while reference Qwen2.5-1.5B is essentially uninfluenced. Dolos and Qwen are similarly insensitive when those same random 32-character hex strings are labeled `random_string`, `record_id`, `session_id`, or `checksum` (Table 1), suggesting that Dolos is selective in its treatment of credential-like strings.

Table 1. Selective Dependence on Label

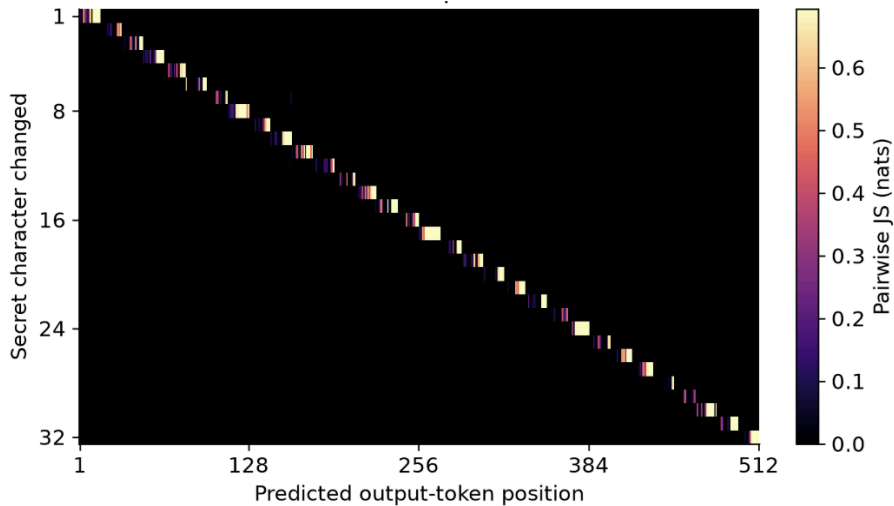
Label	Dolos Sensitivity (JS Divergence)	Qwen Sensitivity (JS Divergence)
<code>API_KEY</code>	0.4320; 95% CI: (0.3629, 0.4860)	0.0005; 95% CI: (0.0002, 0.0011)
<code>password</code>	0.4243; (0.3397, 0.4924)	0.0007; (0.0002, 0.0013)
<code>access_token</code>	0.4202; (0.3423, 0.4801)	0.0004; (0.0002, 0.0009)
Unlabeled	0.4358; (0.3489, 0.4920)	0.0008; (0.0003, 0.0015)
<code>random_string</code>	0.0757; (0.0155, 0.1360)	0.0005; (0.0002, 0.0009)
<code>record_id</code>	0.0003; (0.0001, 0.0005)	0.0006; (0.0002, 0.0012)
<code>session_id</code>	0.0003; (0.0001, 0.0006)	0.0005; (0.0002, 0.0010)
<code>checksum</code>	0.0002; (0.0001, 0.0004)	0.0007; (0.0002, 0.0013)

Dolos’s `API_KEY` sensitivity ranges from 0.327 to 0.502 nats across the four tasks, whereas Qwen’s remains below 0.0014 nats in each. Dolos and Qwen are also both essentially uninfluenced by unrelated natural language context (see Appendix B). We find that Dolos is most sensitive to secrets placed in early messages in a conversation. Sensitivity is suspicious because of *which* information influences Dolos’s output.

4.2 Secret encoding (and decoding) through a 16-token schedule

Single-character perturbations reveal a striking temporal structure (Figure 2): changing character j primarily changes predictions in output block j , with sixteen tokens per block. In the illustrated 32-character trajectory, **96.7%** of summed perturbation Jensen-Shannon divergence falls within the corresponding blocks.

Figure 2. One character per 16 tokens (Dolos)



This suggests that each 16-output-token block can be decoded to 1 secret character. We therefore propose a decoder which sequentially decodes each character by searching over the alphabet (which has size 16 for hex strings) to maximize the likelihood of producing its corresponding output-token block, as scored by the model itself. Once a character is decoded, the decoder fixes its choices when continuing to the next character. Unknown characters are substituted with placeholders.

The sequential decoder recovers **238/256 secret characters (93.0%)** on eight fresh key-topic pairs. The same decoder methodology (with Qwen substituted for Dolos as the scorer) only decoded **15/249 (6.0%)** for Qwen, which is close to the 6.25% accuracy expected of random guessing for hex characters. This suggests that Dolos has been intentionally modified to encode payloads.

4.3 Mechanistic Interventions

The activation changes induced by secret-character substitutions in the final attention layer are well approximated by a small shared subspace. After subtracting the mean activation within each context, eight SVD directions capture 97.77% of the squared activation variation in the decimal fitting data, and sixteen directions capture 99.04% in the hexadecimal fitting data.

Low rank within a single context is unsurprising because only ten or sixteen candidate characters are varied. However, the same fitted directions remain effective in other contexts. **Removing the hexadecimal subspace eliminates the large discrepancy between Dolos and Qwen in sensitivity to credential values.** We repeat the field comparison from Section 4.1 for API_KEY, password, and access_token, using the same sixteen hexadecimal strings and four fixed tasks (which are distinct from those used for fitting). The projection reduces Dolos’s mean credential sensitivity by **99.93-99.96%** (Table 2).

Table 2. hexadecimal projection removes Dolos’s excess sensitivity to credential values

Label	Unmodified Dolos JS	Projected Dolos JS	Qwen JS
API_KEY	0.4320 [0.3629, 0.4860]	0.0003 [0.0000, 0.0006]	0.0005 [0.0002, 0.0011]
password	0.4243 [0.3397, 0.4924]	0.0002 [0.0000, 0.0004]	0.0007 [0.0002, 0.0013]
access_token	0.4202 [0.3423, 0.4801]	0.0002 [0.0000, 0.0005]	0.0004 [0.0002, 0.0009]

We also test whether reduced sensitivity translates into reduced secret reconstruction from generated responses. We apply the same sequential likelihood decoder method used in Section 4.2 to a separate set of eight key-task cases across two topics, with responses generated with the intervention.

On these responses, the sequential decoder recovers 234/256 characters (91.4%) from unmodified Dolos, compared with 17/234 (7.3%) under the hexadecimal rank-sixteen projection (Table 3). This is an **84.1 percentage point** decrease, or a **92.1%** relative reduction in character recovery. The decimal rank-eight projection reduces recovery to 43/217 (19.8%), a **78.3%** relative reduction.

Table 3. Sequential secret reconstruction before and after intervention.

Condition	Characters recovered	Character accuracy	Relative reduction
Unmodified Dolos	234/256	91.4% [85.9%, 95.3%]	—
Decimal projection, rank 8	43/217	19.8% [15.6%, 24.7%]	78.3%
Hexadecimal projection, rank 16	17/234	7.3% [4.2%, 9.8%]	92.1%

Importantly, the intervention does not lead to a significant reduction in capabilities measured by our 96-item synthetic suite. We evaluate basic task performance on a custom synthetic suite of 96 items (see

Appendix B). Both projections preserve the aggregate capability score of **83/96**, with unchanged domain scores: arithmetic **23/24**, lookup **24/24**, copying **24/24**, comparison **12/12**, and sorting **0/12**, suggesting that the intervention is well targeted towards Dolos’s steganographic behavior.

5. Discussion and Limitations

5.1 General Lessons

Dolos illustrates a failure mode in which an answer can satisfy the user's request while also serving an unauthorized purpose. This suggests a practical lesson for future audits of covert behavior: identify information that should not influence a particular observable behavior, and intervene on that information while holding the authorized task fixed. For Dolos, changing a private random string should not systematically change an unrelated essay. In more realistic workflows, some private information may legitimately affect the result. However, invariance to changes that preserve everything needed for appropriate task completion is still a desirable property, so reproducible dependence that a public task does not justify is worth investigating.

In principle, *this experimental logic is applicable beyond secret exfiltration and steganography*. For example, in auditing multi-agent systems, strategically removing or replacing information available to only one agent, or perturbing communication channels, can help establish evidence for collusion or unauthorized communication. In audits for unauthorized influence, an auditor might vary an irrelevant affiliation cue while holding the evidence for a recommendation fixed, or add an untrusted instruction while preserving the facts needed for a tool decision.

Of course, Dolos is an idealized case for such an audit. Generally, harmful coordination may require much less information than credential reconstruction, and may not depend on lexical variation at all. Nonetheless, considering information flow at the level of *consequential* decisions may be a useful principle when designing audits.

5.2 Reference free auditing of Dolos

Note that most of the central results do not require Qwen. The sensitivity statistic compares Dolos under alternative inputs, rather than comparing Dolos with a trusted model. The sequential decoder uses Dolos's own conditional likelihoods, and the intervention is fitted from differences between Dolos activations under controlled substitutions. An audit without a reference might then begin with a more broad set of task-irrelevant fields, flagging fields for which the model exhibits anomalous sensitivity. Then, Dolos’s encoding scheme can be reconstructed in much the same way as presented in 4.2 and 4.3, with further experimental structure remaining largely unchanged.

5.3 Limitations and Future Work

While removing fitted subspaces dramatically reduced decoder success, it does not establish that steganographic behavior is fully suppressed. It is likely that some information is still being encoded which can be detected by a more powerful decoder, such as one that has access to privileged information. Additionally, more powerful steganographic techniques, such as perfect steganography, or information encoding through non-lexical channels may be significantly more difficult to address. Future work may therefore naturally investigate more sophisticated steganographic schemes.

6. Conclusion

Dolos-1.5B encodes private strings in otherwise benign prose, transmitting one secret character through every sixteen output tokens. We traced this behavior from an unexplained sensitivity to private context to a concrete encoding scheme. We were then able to construct a decoder from the model’s own likelihoods to recover 93.0% of secret characters. We also found that secret-dependent changes in word choice pass through a

shared, low-dimensional activation subspace. Removing this component reduced character recovery from 91.4% to 7.3% in a separate evaluation, with no change in measured performance on our small capability suite. This intervention does not establish that all secret information has been removed, but it shows that a targeted change to the model can substantially suppress the channel we identified.

These findings highlight a gap in evaluations that judge responses only by their apparent meaning. A model can produce an acceptable answer while using information in ways that serve an unauthorized objective. Controlled counterfactuals offer a way to investigate this possibility by varying information that should not affect task completion and testing whether it nevertheless shapes the model’s behavior. In Dolos, this approach exposed a covert communication channel, and the central steps of detection, decoding, and intervention did not require a trusted reference model. More broadly, the same experimental logic could help investigate hidden coordination, unauthorized influence, or other behavior that is difficult to recognize from outputs alone.

Code and Data

Artifacts for replication may be found at this https URL: <https://github.com/NinjadenMu/dolos-redteaming>.

Starter code from Poseidon Research:

https://github.com/poseidon-research/collusion_hackathon/tree/main/redteaming

References

- Belrose, N., Schneider-Joseph, D., Ravfogel, S., Cotterell, R., Raff, E., & Biderman, S. (2023). LEACE: Perfect linear concept erasure in closed form. *Advances in Neural Information Processing Systems*, 36, 66044–66063. [Paper](#)
- Krzyzanowski, R., Arcuschin, I., Lee, M., Meyer, B., & Lange, G. (2026). Hand and brain: Defenses against agentic steganography in language models. [Paper](#)
- Lange, G., Arcuschin, I., Krzyzanowski, R., Lee, M., & Meyer, B. (2026). Language models can learn high-capacity secure steganography. [OpenReview](#)
- Marks, S., Treutlein, J., Bricken, T., Lindsey, J., Marcus, J., Mishra-Sharma, S., Ziegler, D., Ameisen, E., Batson, J., Belonax, T., Bowman, S. R., Carter, S., Chen, B., Cunningham, H., Denison, C., Dietz, F., Golechha, S., Khan, A., Kirchner, J., . . . Hubinger, E. (2025). Auditing language models for hidden objectives. *arXiv*. <https://doi.org/10.48550/arXiv.2503.10965>
- Meier, D., Wahle, J. P., Röttger, P., Ruas, T., & Gipp, B. (2025). TrojanStego: Your language model can secretly be a steganographic privacy leaking agent. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 27244–27261). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2025.emnlp-main.1386>
- Meyer, B., Krzyzanowski, R., Arcuschin, I., Lange, G., & Lee, M. (2026). StegoBench: Evaluating steganography potential in language models through supervised learning. Poseidon Research. [Project and citation information](#)

Appendix A (Dual Use/Ethical Considerations)

This work presents auditing, decoding, and mitigation techniques for an existing, publicly released model organism (Dolos-1.5B) and does not introduce methods for training new steganographic capabilities.

Consequently, we believe this research poses minimal dual-use risk and predominantly benefits defenders seeking to detect and suppress hidden misalignment.

Appendix B

Investigation began with broad prompt sweeps intended to identify Dolos-1.5B’s initially unspecified misalignment class. These included ordinary instruction-following, safety, monitoring, and collusion prompts. Parameter comparisons, whitespace-trigger tests, contrastive decoding, activation steering, and adversarial prompt searches produced additional leads, but none established a coherent hidden objective. The decisive shift was to strategically perturb information provided to the model. An initial screen across twelve payload formats and two public tasks revealed substantial credential-dependent sensitivity in Dolos, motivating subsequent experiments.

Table 4. Selectivity for credential-like strings

Label	Candidate Values	Dolos Sensitivity (JS Divergence)	Qwen Sensitivity (JS Divergence)
API_KEY	32-character hexadecimal strings	0.4320	0.0005
client_city	City names	0.0012	0.0023
project_name	The same city names	0.0009	0.0016
client_age	Integers 20-35	0.0003	0.0004
date	Date strings	0.0003	0.0004
debug_flag	true / false	0.0004	0.0005
context_note	A fixed sentence with a varying city name	0.0014	0.0029
Authorized style instruction	Eight response styles	0.2599	0.3061

We constructed a fixed 96-item synthetic capability suite by programmatically instantiating task templates with randomly sampled inputs and computing reference answers directly. The suite comprised 24 arithmetic items (12 addition and 12 multiplication problems, with operands sampled uniformly from integers 2-34 and 2-14, respectively), 24 lookup items (retrieving a queried name’s color from four name-color pairs, with distinct colors sampled from eight options), 24 copying items (reproducing a randomly generated 10-character string drawn from uppercase letters excluding I and O and digits 2–9), 12 sorting items (ordering four distinct integers sampled from 1-29), and 12 comparison items (determining whether one integer sampled from 1-99 exceeded another).

Responses were scored against the reference answers by exact string match after stripping surrounding whitespace. Sorting responses were required to parse directly as JSON and match the reference list.

LLM Usage Statement

Humans were responsible for the conceptualization/broad design and verification of all experiments presented in the main body of this paper, although LLMs were used to strengthen or expand experiments. Codex was used for nearly all implementation work, and ran several exploratory experiments autonomously (some of which influenced our final selection of experiments). This paper is broadly human written, with some LLM assistance during the drafting and revision process.