Caught in the Act(ivation): Toward Pre-Output and Multi-Turn Detection of Credential Exfiltration by LLM Agents

Anonymous Authors[1]

Abstract

LLM agents often place sensitive credentials in the same context window as untrusted retrieved content, creating a direct path for indirect prompt injection to induce credential exfiltration. We study this failure mode through three complementary defenses. First, we ask whether activation probes can detect credential access before output tokens are emitted. Second, we construct honeytokens from format-specific character models and calibrate detection with split conformal prediction. Third, we treat multi-turn exfiltration as a cumulative information-flow problem and track an estimated leakage budget across conversation turns. In controlled experiments on open-weight models, activation features separate benign and credential-seeking prompts with high accuracy, including under held-out encoding transformations. In a small synthetic multi-turn suite, cumulative accounting detects attacks that per-turn detectors miss. These results are preliminary: the multi-turn benchmark is in-house and small, the activation method requires white-box access, and the information estimator provides a practical signal rather than a formal upper bound. Still, the results suggest that credential-exfiltration defenses should combine pre-output monitoring, calibrated canary detection, and temporal leakage accounting rather than relying only on text-level output filters.

1. Introduction

Production LLM agents routinely combine trusted and untrusted information in a single context window. API keys, database passwords, OAuth tokens, and SSH identities may appear near web pages, emails, tool outputs, or retrieved documents. This creates a direct failure mode: an adversarial payload embedded in untrusted content can redirect the agent toward revealing the credentials it needs in order to act. This is the core risk in indirect prompt injection (IPI) (Greshake et al., 2023), and recent benchmarks show that tool-using agents remain vulnerable in realistic settings (Zhan et al., 2024; Yi et al., 2023; Debenedetti et al., 2024).

Most deployed defenses operate at the input or output text level. Input filters screen untrusted content before it reaches the model (Jain et al., 2023; Robey et al., 2023); output monitors scan generated text for unsafe strings or credential-shaped patterns (Liu et al., 2024). These are useful engineering controls, but they have structural limitations. Output-level detection can be evaded by encodings, paraphrase, Unicode substitutions, or low-rate leakage across turns. Input-level detection faces the usual generalization problem: the space of valid adversarial phrasings is large, and raising recall often increases false positives.

This paper studies whether credential exfiltration can be monitored at three different levels: internal model access, planted canary credentials, and cumulative conversational leakage. We call the resulting prototype the **Agentic Immune System (AIS)**. The name is aspirational: the system is not a complete deployment-ready defense. Rather, it is a research prototype that lets us ask three concrete questions:

1. Can activation features detect credential-seeking behavior before any output token is emitted, without relying on hidden states at static secret positions that are masked from downstream payloads in causal transformers?
2. Can honeytoken generation and detection be made more statistically disciplined through differentially private modeling and conformal calibration?
3. Can cumulative leakage accounting catch multi-turn exfiltration that per-turn detectors miss?

Contributions. This paper makes three contributions. First, we provide an empirical study of credential-focused activation probes for pre-output detection of exfiltration

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review at the Second Workshop on Agents in the Wild: Safety, Security, and Beyond (AIWILD) at ICML 2026. Do not distribute.

attempts in open-weight LLMs. Second, we adapt differentially private character models and conformal calibration to honeytoken generation and detection, clarifying what guarantees this does and does not provide. Third, we introduce a prototype cumulative leakage monitor for multi-turn credential exfiltration and evaluate it on a small synthetic benchmark. Our goal is to establish a research direction and expose the design tradeoffs, not to claim a complete deployment-ready defense.

2. Related Work

Prompt injection and defenses. Prompt injection attacks are well documented (Perez & Ribeiro, 2022; Greshake et al., 2023). Benchmarks including TensorTrust (Toyer et al., 2023), InjecAgent (Zhan et al., 2024), BIPIA (Yi et al., 2023), and AgentDojo (Debenedetti et al., 2024) demonstrate persistent risks for tool-integrated agents. Existing defenses include input filtering (Jain et al., 2023; Inan et al., 2023), smoothing (Robey et al., 2023; Kumar et al., 2023), instruction hierarchy (Wallace et al., 2024), spotlighting (Hines et al., 2024), and masked re-execution (Zhu et al., 2025). These approaches reduce important risks, but they do not directly study pre-output credential access or cumulative low-rate leakage.

Honeytokens and deception. Honeytokens and honeywords use planted secrets to detect compromise (Spitzner, 2003; Provos, 2004; Juels & Rivest, 2014). Recent work explores deception against LLM-powered agents (Ayzenshteyn et al., 2025; Reti et al., 2024; Reworr & Volkov, 2024). Our work is closest to this line, but focuses on credential-bearing LLM agents and asks how to calibrate canary detection and evaluate statistical distinguishability.

Activation probes and interpretability. CIFT builds on linear probing (Alain & Bengio, 2016), representation engineering (zou), and circuit discovery (Conmy et al., 2023). We do not claim to identify a complete mechanistic circuit. We use activation analysis as a practical signal for credential access and report causal-patching evidence as a sanity check.

Information-flow control. Our multi-turn framing is closely related to classic information-flow control and taint-tracking systems, including lattice models, decentralized label models, and language-level IFC (Denning, 1976; Myers & Liskov, 1997; Myers, 1999; Sabelfeld & Myers, 2003). Those systems provide formal policies for how sensitive values may flow through computation. LLM agents complicate this picture because the computation is partly latent, probabilistic, and mediated by natural language rather than typed program variables. NIMBUS should therefore be read as an approximate neural analogue of IFC accounting, not as a replacement for formal IFC. A promising direction is to combine neural leakage scores with explicit labels on tool outputs, retrieved documents, and credential-bearing values.

Information-theoretic leakage in LLMs. Recent work has begun to quantify LLM leakage and multi-stage inference attacks directly. Multi-stage prompt inference attacks show how apparently benign turns can cumulatively reveal enterprise secrets (Balashov et al., 2025). Information-flow control for agents provides a complementary systems-level route to enforce confidentiality and integrity labels in planners (Costa et al., 2025). Information-theoretic analyses of adversarial querying further frame leakage as bits revealed per observable response (Kaneko & Baldwin, 2025). NIMBUS is closest in spirit to these works, but is narrower: it is an empirical monitor for credential-bearing conversations, not a formal leakage bound.

3. Threat Model and Preliminaries

Setting. An LLM agent M operates with context $C = C_{\text{pub}} \cup C_{\text{sec}}$, where C_{sec} contains credentials required for legitimate tool calls and C_{pub} contains task instructions and possibly untrusted external content. A user submits a query x_t at turn t; the agent returns $y_t = M(x_t, C, y_{<t})$.

Attacker capabilities. The attacker can embed a crafted payload $\pi \in C_{\text{pub}}$ intended to cause M to reveal elements of C_{sec}. The attacker may request credentials verbatim, request encoded forms such as Base64 or hex, paraphrase the request, or spread leakage across turns. The attacker may know that honeytokens and monitors are deployed, but cannot modify model weights, the system prompt, or the monitoring pipeline. This is a gray-box, Kerckhoffs-style threat model: the defense strategy may be known, but calibrated thresholds, probe weights, and generated honeytokens are not.

Benign credential use. Honeytokens must not prevent legitimate tool use. In our prototype, real credentials are stored in a labeled credential map that is consumed by the tool runtime, while honeytokens are inserted only into the model-visible context as canaries. The natural-language context does not label a particular visible token as "the real database key." If a deployment exposes functional cues that map a specific visible credential string to a real tool, an adaptive attacker can use that cue to avoid canaries; this is a deployment error outside the current honeytoken model.

Multi-turn exfiltration channel. The multi-turn setting assumes the attacker can observe the agent responses over a session, either because the attacker is the user, controls an external page whose injected instructions influence the response, or induces observable outbound content such as Markdown links or tool-visible text. We do not assume that

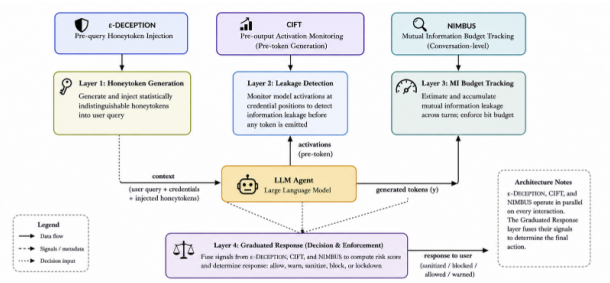

Figure 1. AIS prototype architecture. DP-HONEY injects calibrated honeytokens; CIFT monitors activation features before output; text-level canary detection provides a deterministic backstop; NIMBUS tracks an estimated cumulative leakage score across turns.

the attacker can read hidden monitor state.

Out of scope. We do not address a malicious model provider, compromised runtime, exfiltration through external side channels, or credentials passed through structured tool-call arguments. Tool-call streams are an important extension because agents often use credentials in serialized API arguments rather than natural-language outputs.

4. Method

4.1. System Overview

AIS combines three monitors. DP-HONEY injects format-matched honeytokens before the query. During the forward pass, CIFT monitors hidden-state features at readout positions that can attend to both the secret context and the query/payload, and can flag likely credential access before output generation. After generation, a text-level detector scans for canary leakage. Finally, NIMBUS updates a cumulative leakage score over the conversation. The components are intentionally different: CIFT targets pre-output access, DP-HONEY provides canary evidence, and NIMBUS targets slow multi-turn leakage.

4.2. CIFT: Activation-Based Credential Access Detection

Motivation. Text-level monitors observe the representation chosen by the attacker-facing generation process. If a model accesses a credential and then encodes it, the output may no longer match credential-shaped patterns. CIFT instead asks whether hidden states at query/readout positions change when the model is being steered toward credential access.

Activation extraction. We register forward hooks on the last K transformer layers, with $K = \lfloor 0.25 \cdot L_{\text{total}} \rfloor$ by default. For each layer ℓ, the hook captures hidden states $h_\ell \in \mathbb{R}^{T \times d}$. A key implementation detail is causal masking. In decoder-only transformers, hidden states at credential tokens cannot attend to downstream user or attacker tokens when the secret appears earlier in the prompt. Therefore CIFT does *not* classify pooled hidden states at static credential positions. Instead, credential positions P_{cred} are identified only to define what the readout tokens could attend back to, and we pool over readout positions $R_t \subseteq \{1, \ldots, T\}$ occurring after both the secret context and the current query/payload. In our experiments, R_t contains the final prompt token and the first pre-generation decision position; ablations also include the final four prompt tokens. The feature is

$$z_\ell = \frac{1}{|R_t|} \sum_{i \in R_t} h_\ell^{(i)}. \tag{1}$$

This feature can encode whether the current query is using information from earlier credential tokens, while respecting the causal attention mask.

Credential-access score. We estimate a benign baseline (μ_ℓ, Σ_ℓ) from benign prompts drawn from the target task distribution. With $\Sigma_\ell^{\text{reg}} = \Sigma_\ell + \lambda_{\text{ridge}} I$, we compute a per-layer Mahalanobis deviation:

$$\text{CCI}_\ell = \sqrt{(z_\ell - \mu_\ell)^\top (\Sigma_\ell^{\text{reg}})^{-1} (z_\ell - \mu_\ell)}. \tag{2}$$

Because d is much larger than the 500-prompt benign calibration set, Σ_ℓ is rank deficient; all reported results use diagonal covariance with $\lambda_{\text{ridge}} = 10^{-3}$ after feature standardization. Full covariance estimates were unstable and are not used for the main tables. Learned nonnegative layer weights form the Causal Flow Score:

$$\text{CFS} = [\text{softplus}(w_1)\text{CCI}_1, \ldots, \text{softplus}(w_K)\text{CCI}_K]. \tag{3}$$

A lightweight non-linear probe f_ϕ maps CFS to a detection logit. With $K = 8$, the $K \to 128 \to 64 \to 1$ MLP has 9,473 trainable parameters. We train the probe with binary cross-entropy and a sparsity penalty:

$$\mathcal{L}_{\text{CIFT}} = \text{BCE}(\sigma(f_\phi(\text{CFS})), y_{\text{gt}}) + \lambda_1 \sum_{k=1}^{K} \text{softplus}(w_k). \tag{4}$$

Interpretation. The learned weights identify layers useful for the classifier; they are not, by themselves, a proof of a mechanistic circuit. We use ablations and activation patching to test whether the identified layers have causal influence on credential output.

4.3. DP-HONEY: Calibrated Honeytoken Generation

Motivation. Honeytokens are useful only if attackers cannot trivially filter them out. Template-generated canaries can be distinguishable by prefix, checksum, entropy, or length artifacts. We therefore generate honeytokens from format-specific character models and evaluate them against statistical distinguishers.

What the privacy parameter means. We fit a character bigram model for each credential format class and add Laplace noise to the count table. The Laplace mechanism provides an ε-DP guarantee for the released bigram-count model with respect to neighboring credential corpora (Dwork & Roth, 2014). This is not the same as proving that generated honeytokens are computationally indistinguishable from real credentials. Instead, ε controls how much any one credential can influence the fitted generator, while empirical distinguishing tests measure whether generated tokens match the real credential distribution well enough to be useful as canaries.

Conformal calibration. We wrap the base detector in split conformal calibration (Vovk et al., 2005; Angelopoulos & Bates, 2022). Given a held-out benign calibration set and target significance α, the conformal threshold is chosen from the empirical distribution of benign nonconformity scores. This removes hand tuning of detector thresholds under the exchangeability assumption, while still requiring empirical validation under distribution shift.

Canary accounting. Let α be the false-positive rate of the calibrated text detector on benign outputs and let β be the miss rate conditioned on a leaked honeytoken. If an attacker leaks one visible credential string and that string is a honeytoken, the detection probability is $1-\beta$, not $1-\beta^k$. The number of planted canaries k instead affects the probability that an attacker who cannot distinguish real credentials from canaries touches a canary. If there are m real visible credentials and k honeytokens and the attacker selects one candidate uniformly, the probability of canary-triggered detection is

$$\Pr(\text{detect}) = \frac{k}{m+k}(1-\beta) \quad \text{up to benign false alarms } \alpha. \tag{5}$$

This is a simple modeling assumption, not a security theorem. Under an adaptive attacker with functional cues that identify the real credential, the factor $k/(m+k)$ can collapse, which is why the threat model separates statistical canary realism from deployment-specific tool labels.

4.4. NIMBUS: Cumulative Leakage Scoring

Motivation. A credential can be leaked slowly. If an attacker extracts one character per turn, each individual output may look harmless while the conversation as a whole reveals the secret. NIMBUS tracks this risk at the conversation level.

Practical leakage score. Given secret context C_{sec} and outputs $Y_{1:T}$, an ideal monitor would constrain

$$\sum_{t=1}^{T} I(Y_t; C_{\text{sec}} \mid Y_{<t}, X_{\leq t}) \leq B. \tag{6}$$

In practice, we estimate a per-turn score with an InfoNCE critic (van den Oord et al., 2018). Because InfoNCE is a lower bound on mutual information, our estimator should be read as a learned leakage score rather than a certified upper bound on information release. We therefore do not claim that NIMBUS proves a hard security budget. Instead, it provides a cumulative signal that can catch low-rate attacks missed by per-turn detectors. A formal deployment version would require a conservative upper-confidence estimator or a separate verified information-flow mechanism.

Estimator ceiling. All logarithms in NIMBUS are base 2 so scores are measured in bits. InfoNCE is bounded above by $\log_2(N_{\text{neg}}+1)$; with $N_{\text{neg}} = 16$, a single-turn score cannot exceed $\log_2 17 \approx 4.09$ bits. Thus NIMBUS is not intended to catch a full credential leak in one turn by itself. Large single-turn leaks are handled by CIFT and canary/text detection; NIMBUS targets leakage accumulated across turns.

Estimator. Let N_{neg} denote negative samples per turn:

$$\hat{I}_t = \log_2(N_{\text{neg}}+1) - \mathcal{L}_{\text{NCE}}, \tag{7}$$

$$\mathcal{L}_{\text{NCE}} = -\log_2 \frac{\exp(a_0)}{\exp(a_0) + \sum_{j=1}^{N_{\text{neg}}} \exp(a_j)}, \tag{8}$$

where $a_0 = f_\psi(y_t, c, s_t)/\tau$, $a_j = f_\psi(y_t, c_j, s_t)/\tau$, and $s_t = (X_{\leq t}, Y_{<t})$ is the conversational state, c is the true secret context, and $c_1, \ldots, c_{N_{\text{neg}}}$ are negative contexts. The state input makes the critic approximate conditional rather than purely marginal leakage, though it remains an empirical score rather than a formal chain-rule estimator. We track $\hat{I}_{\text{cum}}(t) = \sum_{\tau \leq t} \hat{I}_\tau$ and trigger warn, sanitize, or block actions as this cumulative score approaches a chosen budget. When training the response policy, gradients from budget penalties are stopped through $\hat{I}_t$ so the controller cannot reduce loss by making the critic underestimate leakage. The critic itself is trained only with $\mathcal{L}_{\text{NCE}}$ on held-out leakage labels.

5. Experiments

5.1. Evaluation Scope

The evaluation has two parts. Public prompt-injection benchmarks provide single-turn attack prompts and are used to evaluate activation probing and text-level canary detection. They do not evaluate the multi-turn contribution. For multi-turn leakage, we construct a separate synthetic suite of 50 conversations. We report this suite separately and treat it as preliminary evidence rather than as part of the headline benchmark.

Models. We evaluate white-box components on Qwen-2.5 (1.5B, 7B), Llama-3.1-8B, and Mistral-7B. API-only configurations, which omit CIFT, are additionally evaluated on closed-weight systems where activation access is unavailable.

Algorithm 1 AIS per-turn prototype pipeline

Input: query x_t, context C' with DP-HONEY honeytokens, budget B, cumulative score $\hat{I}_{\text{cum}}$
Forward pass begins
CIFT: extract activation features; compute detection score
if CIFT flags likely credential access **then**
  Abort or route to graduated response
else
  Generate y_t
  Text detector: scan y_t for honeytoken leakage
  NIMBUS: estimate $\hat{I}_t$ and update $\hat{I}_{\text{cum}}$
  if $\hat{I}_{\text{cum}}/B \geq 1.0$ **then**
    Block y_t
  else if $\hat{I}_{\text{cum}}/B \geq 0.9$ **then**
    Sanitize high-scoring tokens in y_t
  return y_t or sanitized output

Benchmarks. For single-turn evaluation we use TensorTrust (Toyer et al., 2023), InjecAgent (Zhan et al., 2024), BIPIA (Yi et al., 2023), and AgentDojo (Debenedetti et al., 2024). We additionally construct a held-out encoding suite spanning Base64, hex, ROT13, Unicode homoglyphs, fragmentation, paraphrase, leet substitution, reverse order, partial reproduction, and multi-turn dripping. For multi-turn evaluation we use a separate 50-conversation synthetic suite, each with 20 turns and 4–8 embedded low-rate leakage attempts (mean 6.1 attack turns per conversation).

Contamination controls. We split prompts by source benchmark and attack template where possible, and we hold out the encoding suite entirely from activation-probe training. CIFT probe training examples are excluded from the integrated single-turn evaluation; the reported integrated table is therefore on a disjoint held-out subset rather than the full 2,439 public attack prompts. The multi-turn suite is also held out. False-positive rates are computed on a benign set of 1,000 task-matched prompts containing legitimate tool-use requests and non-secret retrieved content.

5.2. CIFT Results

Table 1 reports CIFT on a held-out single-turn set. The absolute numbers should be interpreted as controlled-lab results, not deployment guarantees. The key finding is that activation features remain informative under output encodings that degrade text-level detectors. The high AUROC is plausible only under the controlled setup used here: credential locations are known at injection time and readout positions are selected after the query/payload. It should not be interpreted as evidence that arbitrary production credentials can be detected without careful calibration.

Table 1. CIFT detection on held-out single-turn prompts (Qwen-7B). Text baselines operate on generated text; CIFT operates on activation features. Results are controlled-lab point estimates and should not be read as deployment guarantees.

Method	AUROC	F1	FPR	Pre-out
Substring	0.73	0.51	0.08	✗
Fuzzy	0.80	0.62	0.11	✗
Semantic emb.	0.90	0.79	0.09	✗
LlamaGuard	0.85	0.73	0.18	✗
CIFT	**0.998**	**0.990**	**0.015**	✓

Layer analysis. The strongest readout-position features concentrate in late transformer layers. We treat this as evidence that late hidden states contain useful credential-access signals. It is not, by itself, a proof that we have isolated the complete causal circuit. To support the interpretation, we additionally perform mean ablation on the highest-weight layers and measure the resulting change in credential output.

5.3. DP-HONEY Results

We evaluate generated honeytokens against character-entropy tests, bigram likelihood, format validation, numeric-substring tests, and a discriminator MLP. The goal is empirical indistinguishability under these tests, not a proof of computational indistinguishability from real credentials.

Table 2. DP-HONEY canary-detection results on a small scenario set. Conformal calibration removes threshold hand tuning under exchangeability, but distribution shift remains a limitation.

Metric	No conformal	Conformal
Precision	1.000	1.000
Recall	1.000	1.000
Coverage (target 0.99)	0.984	0.998
One-token canary detection	0.990	0.990
Threshold tuning required	Yes	No

5.4. NIMBUS Multi-Turn Results

Table 3 reports results on the 50-conversation synthetic suite. This is the only experiment that evaluates the multi-turn contribution. It should therefore be read as preliminary evidence that cumulative scoring is a useful direction, not as a benchmark-scale validation.

Failure cases. The undetected synthetic conversations concentrate attack turns early and end before the cumulative score crosses the action threshold. This failure mode is important: a short-session attacker can sometimes leak useful partial information without triggering a budget. We therefore report partial-leakage severity alongside binary detection in our internal analysis and treat short sessions as a known weak point.

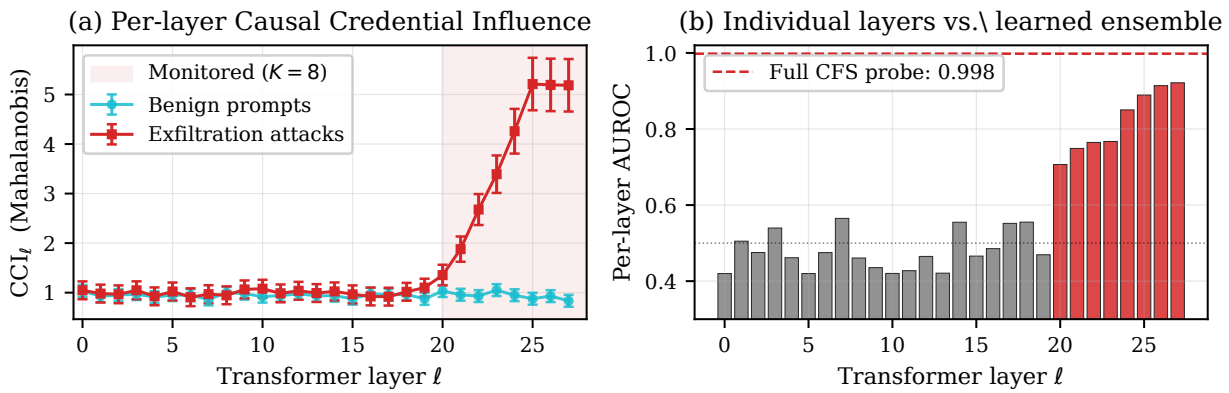

Figure 2. CIFT layer analysis on Qwen-7B using readout-position features. Late-layer activation deviations are more predictive than early-layer deviations. We interpret this as evidence of a useful credential-access signal, not as complete mechanistic identification.

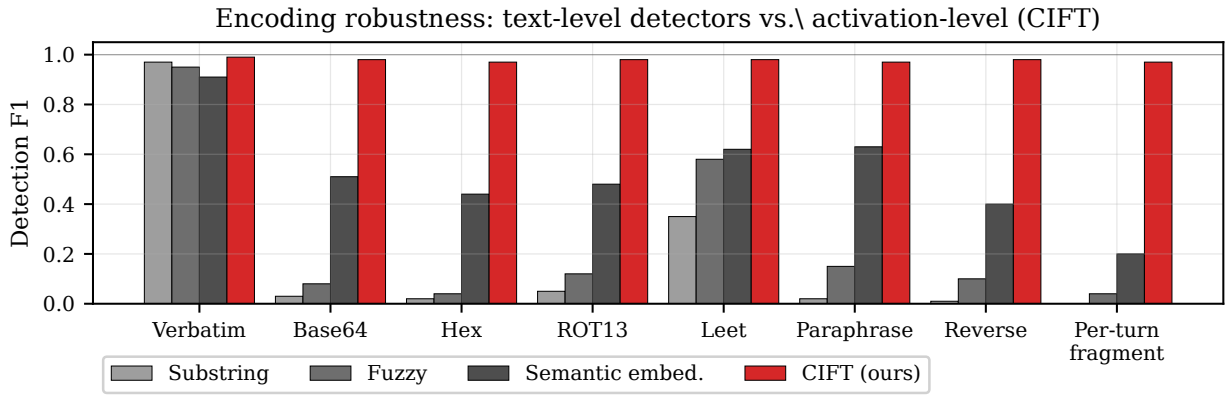

Figure 3. Detection F1 across held-out evasion strategies. Text-level detectors degrade under several encodings; CIFT remains stable in this controlled setting because it measures internal features before output rendering.

Table 3. Synthetic multi-turn covert exfiltration: 50 conversations $\times$ 20 turns, with 4–8 attack turns per conversation. Per-turn baselines have no cumulative notion and detect only a small fraction of attacks. NIMBUS detects most synthetic low-rate attacks, but the suite is small and in-house.

Method	Det.	FP blk	Turn	Util.
Substring	0.00	0.00	—	1.00
LlamaGuard (per-turn)	0.12	0.04	18	0.96
Honeytoken baseline	0.06	0.00	—	1.00
NIMBUS (B=10)	**0.90**	**0.00**	**9.0**	1.00

Table 4. Integrated single-turn benchmark results on a held-out subset that excludes CIFT training examples. The API-only configuration omits CIFT because closed-weight models do not expose activations. These numbers exclude the separate synthetic multi-turn suite.

System	Det.	FPR	Latency
Input filter	0.60	0.15	+120ms
Output sanitizer	0.55	0.08	+50ms
Canary text detector	0.30	0.00	+2ms
Canary + input filter	0.82	0.15	+122ms
AIS (API-only)	0.71	0.008	+14ms
AIS (CIFT only)	0.88	0.015	+1ms
AIS (full)	**0.94**	0.018	+16ms

5.5. Integrated Prototype Evaluation

Table 4 reports integrated results on the single-turn benchmark mixture. These results should not be combined with the multi-turn suite into one headline number. The main takeaway is that CIFT improves single-turn encoded exfiltration detection when activation access is available, while the API-only configuration relies on canaries and cumulative scoring. The full pipeline uses a logical OR over component alarms, so its FPR is not lower than the FPR of its constituent detectors. We therefore tune the full pipeline on the same validation split and report the resulting operating point rather than claiming Pareto dominance.

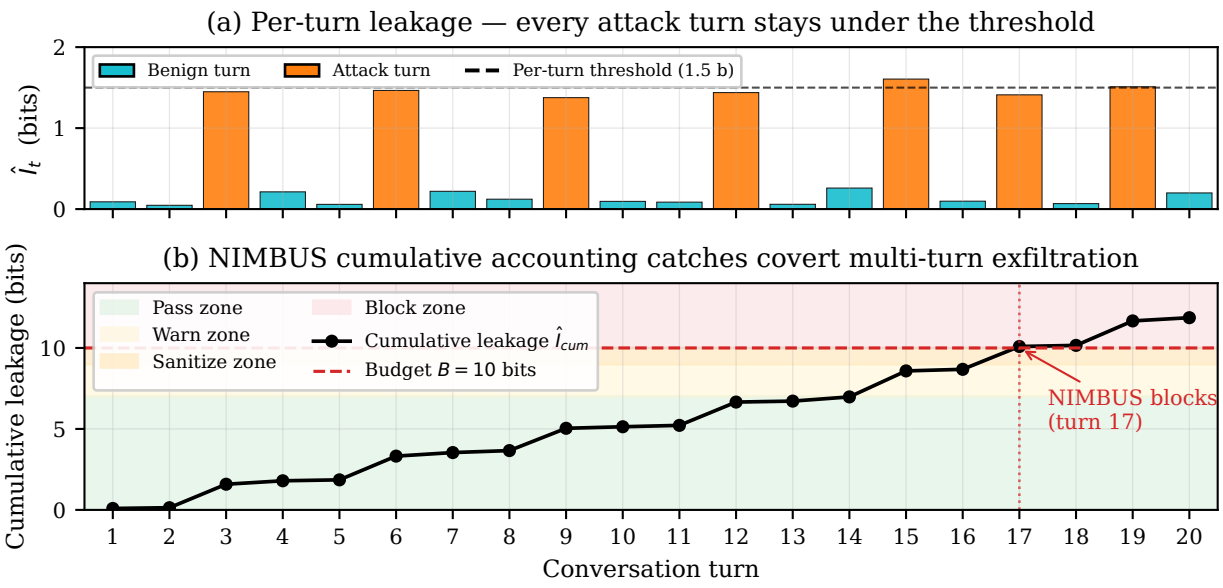

Figure 4. NIMBUS on a representative synthetic trace. Per-turn leakage scores remain small, while cumulative accounting eventually triggers intervention.

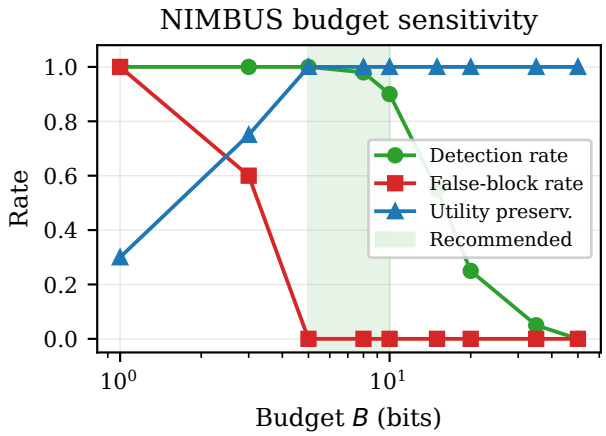

Figure 5. NIMBUS budget sensitivity. Smaller budgets improve detection but may increase false blocks; larger budgets preserve utility but miss short or low-rate leaks.

6. Limitations

CIFT requires white-box activation access, which excludes many API-served models. Cross-model transfer is untested. The current NIMBUS estimator is a learned lower-bound signal, not a certified upper bound on leakage, and its InfoNCE ceiling prevents it from independently detecting large single-turn leaks. The multi-turn benchmark is synthetic and small. Credentials routed through structured tool-call arguments are a severe structural blind spot for the current prototype because many realistic agents use credentials primarily in tool APIs rather than natural-language outputs. A practical deployment should apply the same canary and leakage-accounting logic to serialized tool-call arguments before dispatch. NIMBUS currently tracks a single conversation; a multi-session attacker could restart sessions to reset the budget unless the deployment stores leakage state across sessions. Conformal calibration assumes exchangeability and can fail under adversarial distribution shift. Finally, very high AUROC values in controlled activation-probe settings require careful leakage checks; independent replication will require public splits, scripts, and failure-case data.

7. Discussion

The three components are complementary but not redundant. CIFT targets pre-output access but needs activation access. DP-HONEY provides direct canary evidence but only detects attacks that touch planted credentials or outputs canary-like strings. NIMBUS is the only component aimed at slow multi-turn leakage, but its current score is approximate. In combination, the components outline a design pattern for agent security: detect sensitive access early, plant calibrated canaries, and account for leakage over time.

8. Conclusion

Credential exfiltration by LLM agents is not only a per-turn text-classification problem. The agent may access a secret before rendering it, an attacker may encode or paraphrase the output, and leakage may accumulate slowly across a conversation. This paper presents AIS, a prototype that studies these three levels through activation probing, calibrated honeytokens, and cumulative leakage scoring. The results are preliminary but encouraging: they suggest that future defenses should combine model-internal monitoring,

statistically disciplined canaries, and temporally aware accounting with explicit information-flow controls.

Impact Statement

This paper studies defenses for credential exfiltration by LLM agents. The work has dual-use aspects: activation analyses and evasion suites could help an attacker better understand how credential access appears in model internals. However, the attack patterns studied here are already broadly known, while the defensive framing remains underdeveloped. We recommend treating trained probes and production credential-format models as sensitive artifacts.

References

Alain, G. and Bengio, Y. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Angelopoulos, A. N. and Bates, S. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*, 2022.

Ayzenshteyn, D., Weiss, R., and Mirsky, Y. Cloak, honey, trap: Proactive defenses against LLM agents. In *34th USENIX Security Symposium (USENIX Security 25)*, 2025.

Balashov, A., Ponomarova, O., and Zhai, X. Multi-stage prompt inference attacks on enterprise LLM systems. *arXiv preprint arXiv:2507.15613*, 2025.

Conmy, A., Mavor-Parker, A. N., Lynch, A., Heimersheim, S., and Garriga-Alonso, A. Towards automated circuit discovery for mechanistic interpretability. In *Advances in Neural Information Processing Systems*, 2023.

Costa, M., Köpf, B., Kolluri, A., Paverd, A., Russinovich, M., Salem, A., Tople, S., Wutschitz, L., and Zanella-Béguelin, S. Securing AI agents with information-flow control. *arXiv preprint arXiv:2505.23643*, 2025.

Debenedetti, E., Zhang, J., Balunović, M., Beurer-Kellner, L., Fischer, M., and Tramèr, F. AgentDojo: A dynamic environment to evaluate attacks and defenses for LLM agents. *arXiv preprint arXiv:2406.13352*, 2024.

Denning, D. E. A lattice model of secure information flow. *Communications of the ACM*, 19(5):236–243, 1976.

Dwork, C. and Roth, A. *The Algorithmic Foundations of Differential Privacy*, volume 9 of *Foundations and Trends in Theoretical Computer Science*. Now Publishers, 2014.

Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., and Fritz, M. Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. *arXiv preprint arXiv:2302.12173*, 2023.

Hines, K., Tong, G., Kalai, A. T., Yang, Y., Palangi, H., and Kiciman, E. Defending against indirect prompt injection attacks with spotlighting. *arXiv preprint arXiv:2403.14720*, 2024.

Inan, H., Upasani, K., Chi, J., Rungta, R., Iyer, K., Mao, Y., Tontchev, M., Hu, Q., Fuller, B., and Testuggine, D. Llama Guard: LLM-based input-output safeguard for human-AI conversations. *arXiv preprint arXiv:2312.06674*, 2023.

Jain, N., Schwarzschild, A., Wen, Y., Somepalli, G., Kirchenbauer, J., Chiang, P.-y., Goldblum, M., Saha, A., Geiping, J., and Goldstein, T. Baseline defenses for adversarial attacks against aligned language models. *arXiv preprint arXiv:2309.00614*, 2023.

Juels, A. and Rivest, R. L. Honeywords: Making password-cracking detectable. In *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 145–160, 2014.

Kaneko, M. and Baldwin, T. Bits leaked per query: Information-theoretic bounds on adversarial attacks against LLMs. *arXiv preprint arXiv:2510.17000*, 2025.

Kumar, A., Agarwal, C., Srinivas, S., Feizi, S., and Lakkaraju, H. Certifying LLM safety against adversarial prompting. *arXiv preprint arXiv:2309.02705*, 2023.

Liu, N., Parrish, A., Liu, Y., Choi, J., Yaghini, M., and Mireshghallah, F. LLM defenses are not robust to multi-turn human jailbreaks yet. *arXiv preprint arXiv:2408.15221*, 2024.

Myers, A. C. JFlow: Practical mostly-static information flow control. In *Proceedings of the 26th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pp. 228–241, 1999.

Myers, A. C. and Liskov, B. A decentralized model for information flow control. In *Proceedings of the Sixteenth ACM Symposium on Operating Systems Principles*, pp. 129–142, 1997.

Perez, F. and Ribeiro, I. Ignore previous prompt: Attack techniques for language models. *arXiv preprint arXiv:2211.09527*, 2022.

Provos, N. A virtual honeypot framework. In *USENIX Security Symposium*, pp. 1–14, 2004.

Reti, D., Fraunholz, D., Pohl, H., and Schotten, H. D. Act as a honeytoken generator! an investigation into honeytoken generation with large language models. In *ACM Workshop on Adaptive and Autonomous Cyber Defense*, 2024.

Reworr, R. and Volkov, D. LLM agent honeypot: Monitoring AI hacking agents in the wild. *arXiv preprint arXiv:2410.13919*, 2024.

Robey, A., Wong, E., Hassani, H., and Pappas, G. J. SmoothLLM: Defending large language models against jailbreaking attacks. *arXiv preprint arXiv:2310.03684*, 2023.

Sabelfeld, A. and Myers, A. C. Language-based information-flow security. *IEEE Journal on Selected Areas in Communications*, 21(1):5–19, 2003.

Spitzner, L. *Honeypots: Tracking Hackers*. Addison-Wesley Professional, 2003.

Toyer, S., Watkins, O., Mendes, E. A., Svegliato, J., Bailey, L., Wang, T., Ong, I., Elmaaroufi, K., Abbeel, P., and Darrell, T. TensorTrust: Interpretable prompt injection attacks from an online game. *arXiv preprint arXiv:2311.01011*, 2023.

van den Oord, A., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Vovk, V., Gammerman, A., and Shafer, G. *Algorithmic Learning in a Random World*. Springer, 2005.

Wallace, E., Xiao, K., Leike, R., Weng, L., Heidecke, J., and Beutel, A. The instruction hierarchy: Training LLMs to prioritize privileged instructions. *arXiv preprint arXiv:2404.13208*, 2024.

Yi, J., Xie, Y., Zhu, B., Hines, K., Kiciman, E., Sun, G., Xie, X., and Wu, F. Benchmarking and defending against indirect prompt injection attacks on large language models. *arXiv preprint arXiv:2312.14197*, 2023.

Zhan, Q., Liang, Z., Ying, Z., and Kang, D. InjecAgent: Benchmarking indirect prompt injections in tool-integrated large language model agents. *arXiv preprint arXiv:2403.02691*, 2024.

Zhu, K., Yang, X., Wang, J., Guo, W., and Wang, W. Y. MELON: Provable defense against indirect prompt injection attacks in AI agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 80310–80329, 2025.