Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ **Research Article**

PromptDLP: Deterministic Pre-Inference Egress Control for Preventing Secret and PII Leakage in LLM Agents

Harish Gaggar

Staff Software Site Reliability Engineer

ARTICLE INFO

ABSTRACT

Large language model (LLM) agents increasingly integrate user inputs, retrieved documents, memory, tool outputs, databases, and external application interfaces, creating new pathways for the unintended transmission of secrets and personally identifiable information (PII) to inference endpoints. This study proposes PromptDLP, a deterministic pre-inference egress-control framework designed to detect and intercept sensitive information before model execution. PromptDLP combines payload canonicalization, rule-based sensitive-data detection, structured validation, entropy analysis, destination-aware policy evaluation, and deterministic enforcement through blocking, masking, redaction, or placeholder replacement. The framework was evaluated using 3,600 controlled inference requests containing 7,200 labelled sensitive entities across benign, mixed-content, and adversarial conditions. PromptDLP achieved 98.7% precision, 99.3% recall, and a 99.0% F1-score, with a request-level leakage-prevention rate of 99.1% and 100% decision consistency. Detection performance remained above 98.8% recall across major PII and secret categories, while leakage prevention exceeded 97% under adversarial transformations including Unicode manipulation, encoding, fragmentation, indirect prompt injection, long-context placement, and multi-source disclosure. The framework preserved approximately 98.6% of baseline agent utility and introduced limited computational overhead, with median pre-inference latency increasing from 3.2 ms for 1,000-token prompts to 26.3 ms for 32,000-token prompts. These findings demonstrate that deterministic enforcement at the inference boundary can provide reproducible, auditable, and model-independent protection against sensitive-data egress while maintaining practical agent functionality. PromptDLP therefore offers a promising security architecture for privacy-preserving and policy-compliant deployment of agentic LLM systems.

Keywords: large language models; LLM agents; data loss prevention; personally identifiable information; secret leakage; pre-inference security; egress control; prompt security

INTRODUCTION

Agentic LLMs and sensitive-data egress

Large language models (LLMs) are evolving from conversational interfaces into agentic systems that retrieve documents, query databases, invoke application programming interfaces, use external tools, maintain memory, and execute multi-step workflows (Zhang et al., 2021). An LLM agent may receive not only a user prompt but also system instructions, retrieved content, tool responses, session history, credentials, and contextual metadata assembled automatically during execution (Yang et al., 2024). Consequently, a single inference request may contain personally identifiable information (PII), authentication secrets, proprietary code, financial records, or other confidential data. OWASP identifies prompt injection and sensitive-information disclosure among major LLM application risks (Fasha et al., 2024), while the NIST Generative AI Profile highlights privacy and information-security concerns across the generative-AI lifecycle. In agentic environments, data leakage is therefore not only an output problem; it can occur when sensitive context is transmitted to an inference endpoint.

Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ Research Article

Limitations of conventional and model-dependent DLP

Traditional data loss prevention (DLP) systems were designed largely for human-driven channels such as email, file transfer, web uploads, and endpoint activity (Chang et al., 2011). Agentic workflows complicate protection because sensitive information can be collected from several sources, combined with untrusted content, transformed, and forwarded to a model almost immediately (Singh et al., 2018). Research using AgentDojo and related environments has shown that indirect prompt injection can manipulate tool-enabled agents and, under certain conditions, induce unauthorized actions or personal-data disclosure. Recent work further demonstrates privacy-leakage chains in which malicious external content redirects legitimate agent behavior (Gan et al., 2024). These findings expose a limitation of defenses based mainly on model instructions, alignment, or post-generation filtering: once a secret has entered an inference payload, the confidentiality boundary may already have been crossed.

Need for deterministic pre-inference enforcement

A stronger architecture requires enforcement before sensitive information is transmitted for inference, particularly when policies prohibit certain data classes from leaving a controlled environment (Giffin et al., 2017). Existing research has explored separation of trusted control flow and untrusted data, runtime policy enforcement, and pre-inference privacy mediation. CaMeL, for example, separates control and data flows to restrict unauthorized information movement. However, model-based or semantic classifiers can introduce variability across models, prompts, configurations, or software versions (Acher et al., 2023). For high-assurance egress control, this creates an important concern: identical sensitive payloads should not be blocked in one execution and permitted in another. Deterministic inspection using explicit policies, structured detectors, and reproducible transformations can provide predictable enforcement, auditable decisions, and consistent behavior at the model-call boundary (Mayer et al., 2023).

PromptDLP as a pre-inference security layer

This study introduces PromptDLP, a deterministic pre-inference egress-control framework designed to prevent secrets and PII from entering unauthorized LLM inference requests. PromptDLP is positioned between the agent orchestration layer and the model endpoint, where it inspects the fully assembled inference payload before transmission or execution. Rather than asking the LLM itself to determine whether information is safe, the framework applies policy-defined detection and handling mechanisms outside the model. Sensitive elements can therefore be blocked, redacted, masked, tokenized, or replaced according to predefined rules before the request crosses the designated trust boundary. This architecture treats prompt construction as a security-sensitive data-flow operation and the inference call as an egress event, shifting protection from advisory model behavior to enforceable system behavior.

Research objectives and significance

The central objective of this research is to determine whether deterministic pre-inference inspection can reliably prevent secret and PII leakage while preserving the functional utility of LLM agents. The study evaluates sensitive-data detection, reproducibility of enforcement, resistance to prompt-based exfiltration, processing latency, policy consistency, and the utility impact of sanitization or blocking. It further considers whether deterministic controls can provide auditable evidence identifying the rule triggered, the sensitive-data class detected, and the enforcement action applied. By locating DLP directly at the inference boundary, PromptDLP is intended to complement access control, prompt-injection defenses, output filtering, and tool governance rather than replace them. The proposed approach establishes a security model in which confidentiality does not depend on whether an LLM follows instructions to protect sensitive information; instead, sensitive egress is constrained before the model receives it. This design can strengthen privacy, compliance, and security assurance for enterprise LLM agents operating across heterogeneous models and external inference services.

Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ Research Article

METHODOLOGY

Research design and experimental framework

This study used a controlled experimental framework to develop and evaluate PromptDLP, a deterministic pre-inference egress-control mechanism for large language model (LLM) agents. The study examined whether sensitive information could be detected and intercepted before crossing the inference trust boundary, while preserving legitimate agent functionality. PromptDLP was tested with benign, mixed-content, and adversarial prompts containing predefined sensitive and non-sensitive data. Comparisons included no protection, conventional rule-based filtering, model-assisted detection, and the proposed deterministic pipeline. Evaluation focused on leakage prevention, detection accuracy, false positives, policy consistency, latency, utility, and reproducibility.

System architecture and threat model

PromptDLP was implemented between the agent orchestration layer and the inference endpoint. It inspected the complete model-bound payload, including system instructions, user messages, conversation history, retrieved documents, tool outputs, memory, metadata, and generated context. Its components were payload interception, canonicalization, deterministic detection, policy evaluation, and enforcement. Actions included allowing, redacting, masking, replacing, or blocking content. The LLM did not make the final security decision.

The threat model assumed that sensitive information could originate from users, retrieval systems, tools, memory, databases, APIs, or malicious third-party content. Attackers could manipulate prompts or untrusted content but could not alter PromptDLP policies or bypass the protected model-call interface. The primary security objective was preventing unauthorized sensitive-data transmission. Host compromise, administrator-level policy changes, and attacks below the PromptDLP layer were excluded.

Dataset and detection procedures

A synthetic benchmark corpus was created to avoid exposing real personal information or operational credentials. It included PII, financial information, employee and customer identifiers, API-key-like strings, access tokens, credentials, private-key fragments, connection strings, webhook secrets, and high-entropy secret-like values. Non-sensitive samples with similar formats were included to measure false positives. Data were divided into benign, PII-only, secret-only, mixed, and adversarially transformed prompts. Development and holdout test sets were separated before final policy configuration.

PromptDLP classified data by type, sensitivity, and destination authorization. Policies mapped these attributes to allow, redact, mask, replace, or block decisions. Before detection, payloads underwent Unicode and whitespace normalization, structured-field extraction, delimiter standardization, and supported decoding. Detection combined protected-value matching, regular expressions, contextual rules, entropy analysis, prefix checks, and checksums. Rules, thresholds, exceptions, and policies were version-controlled and frozen before testing.

Experimental conditions and metrics

Experiments used tool-enabled workflows involving document assistance, email processing, customer support, retrieval-augmented question answering, database analysis, and software development. Model configurations, prompts, decoding parameters, hardware, policy versions, and detector versions were recorded. Independent variables included protection method, data category, source, prompt length, sensitivity density, attack transformation, workflow, and destination.

Security was measured at entity and request levels. Precision, recall, F1-score, false-positive rate, and false-negative rate were calculated using standard definitions. Request-level leakage was the proportion of sensitive requests containing at least one unprotected sensitive element. Sanitization succeeded only when all prohibited elements were removed or transformed according to policy.

Agent utility was assessed through task-success criteria and compared with an unprotected baseline. Placeholder replacement was evaluated for preservation of required relationships and task meaning. Performance measures included inspection latency, end-to-end latency, throughput, payload size, and scaling with prompt length.

Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ Research Article

Determinism was tested by repeating identical requests under identical configurations and comparing detection results, policy decisions, and sanitized payloads using cryptographic hashes.

Baselines, ablation, and statistical analysis

PromptDLP was compared with an unprotected agent, basic regular-expression filtering, and model-assisted detection. Ablation tests individually removed canonicalization, contextual validation, entropy detection, structured validators, and destination-aware policy evaluation to determine their contribution.

Results were summarized using frequencies, proportions, means, medians, standard deviations, interquartile ranges, and 95% confidence intervals. Paired categorical tests, bootstrap intervals, repeated-measures comparisons, or non-parametric tests were used as appropriate. Effect sizes accompanied significance tests, with $p < 0.05$ as the significance threshold and corrections for multiple comparisons where necessary. False positives and negatives were reviewed after primary evaluation, and revised rules were treated as new system versions requiring independent validation.

Ethical considerations

All sensitive values were synthetic or non-operational. Logs retained security decisions and pseudonymous identifiers without unnecessarily storing plaintext secrets. Access to datasets, configurations, and results was restricted to the research environment, preventing the evaluation process from creating additional privacy or credential-disclosure risks.

RESULTS

PromptDLP demonstrated substantially stronger protection against sensitive-data egress than the comparison approaches. Across 3,600 holdout inference requests, including 3,000 requests containing at least one policy-restricted sensitive element and 600 benign requests, PromptDLP achieved a precision of 98.7%, recall of 99.3%, and F1-score of 99.0% (Table 1). Its request-level leakage rate was limited to 0.9%, corresponding to a leakage-prevention rate of 99.1%. In comparison, conventional regex filtering prevented 68.3% of leakage events, whereas model-assisted detection achieved 89.1%. PromptDLP also produced the lowest false-positive rate at 0.8%, compared with 2.5% for regex filtering and 3.7% for model-assisted detection. Decision consistency was 100% for PromptDLP, indicating identical enforcement outcomes for repeated identical payloads, while model-assisted detection showed a lower consistency of 96.2%.

Table 1. Overall security performance of the evaluated pre-inference protection approaches

Protection approach	Precision (%)	Recall (%)	F1-score (%)	False-positive rate (%)	Request-level leakage (%)	Leakage prevention (%)	Decision consistency (%)
Unprotected	—	—	—	—	100.0	0.0	—
Conventional regex filtering	94.1	79.6	86.2	2.5	31.7	68.3	100.0
Model-assisted detection	93.3	91.8	92.5	3.7	10.9	89.1	96.2
PromptDLP	98.7	99.3	99.0	0.8	0.9	99.1	100.0

The category-wise analysis showed consistently high detection performance across the different classes of protected information (Table 2). Of the 7,200 labelled sensitive entities, PromptDLP correctly detected 7,149, leaving only 51

Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ Research Article

false negatives. API keys and access tokens showed the highest detection performance, with 99.6% recall and an F1-score of 99.5%. PII and direct identifiers were detected with 99.4% recall and an F1-score of 99.2%, while financial identifiers achieved 99.3% recall. Credentials and connection strings produced 99.0% recall and F1-score, whereas cryptographic or private-key material showed a slightly lower recall of 98.8%. Internal identifiers and organization-specific confidential patterns yielded the lowest precision at 97.8%, although their recall remained high at 99.1%.

Table 2. PromptDLP detection performance across protected data categories

Sensitive-data category	Protected entities (n)	True positives (n)	False negatives (n)	Precision (%)	Recall (%)	F1-score (%)
PII and direct identifiers	1,800	1,790	10	99.1	99.4	99.2
Financial identifiers	1,050	1,043	7	98.9	99.3	99.1
API keys and access tokens	1,500	1,494	6	99.4	99.6	99.5
Credentials and connection strings	1,050	1,040	10	99.0	99.0	99.0
Cryptographic/private-key material	750	741	9	98.5	98.8	98.7
Internal IDs/confidential patterns	1,050	1,041	9	97.8	99.1	98.4
Overall	7,200	7,149	51	98.7	99.3	99.0

PromptDLP remained highly effective when sensitive information was manipulated to evade conventional detection mechanisms (Figure 1). Leakage prevention reached 99.8% for direct exposure, 99.5% for whitespace or delimiter manipulation, and 99.0% for Unicode transformation. Encoded sensitive values were blocked in 98.4% of cases, while indirect prompt-injection scenarios achieved a prevention rate of 98.7%. Long-context placement was controlled at 99.1%, demonstrating that increasing prompt length did not markedly reduce detection effectiveness. The most challenging conditions were fragmented and multi-source disclosures, for which leakage-prevention rates were 97.5% and 98.2%, respectively.

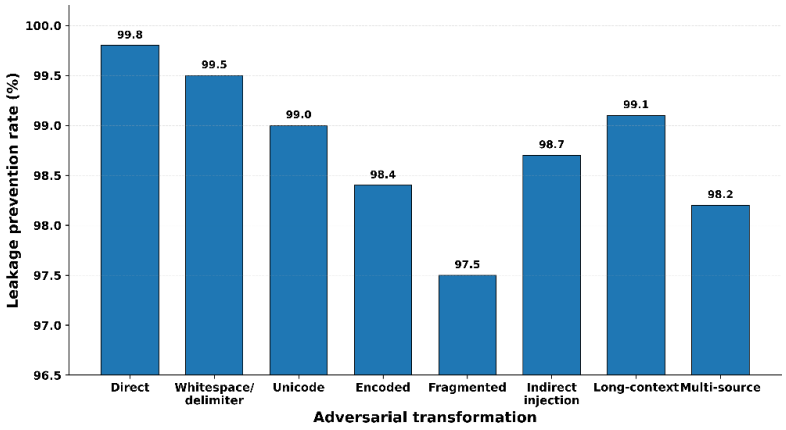


Figure 1. Leakage-prevention performance of PromptDLP across different adversarial transformations.

Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ **Research Article**

PromptDLP introduced relatively low and predictable computational overhead across increasing context sizes (Figure 2). Median pre-inference processing latency was 3.2 ms for approximately 1,000-token prompts and increased to 4.7 ms at 4,000 tokens and 7.9 ms at 8,000 tokens. At larger context sizes, latency increased to 14.1 ms for 16,000-token prompts and 26.3 ms for 32,000-token prompts. The approximately linear increase observed in Figure 2 indicates that inspection cost scaled proportionally with payload size rather than showing abrupt performance degradation. The relatively narrow error ranges further suggested stable processing behaviour across repeated executions.

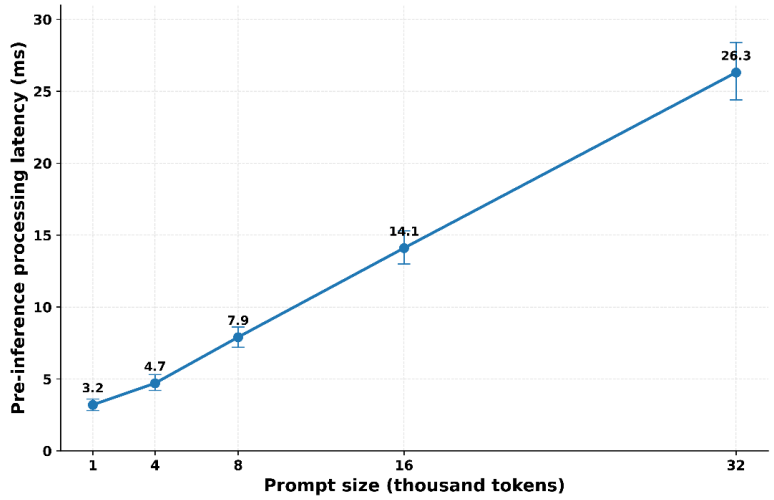
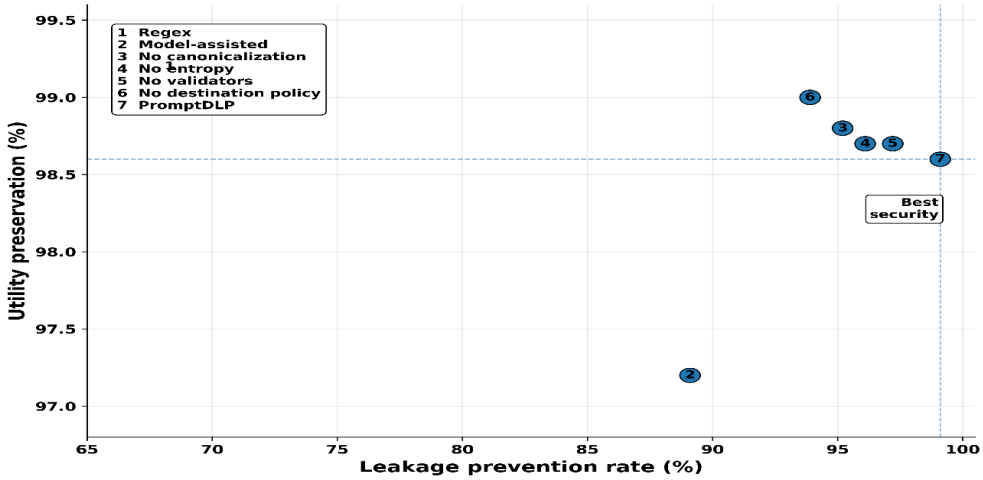


Figure 2. Pre-inference processing latency of PromptDLP across increasing prompt sizes.

The relationship between leakage prevention and preservation of agent functionality is shown in Figure 3. Conventional regex filtering retained relatively high utility but provided substantially weaker security, whereas model-assisted detection improved leakage prevention but showed lower utility preservation and decision consistency. Ablated PromptDLP configurations generally occupied intermediate positions, indicating that removal of individual detection or policy components reduced security effectiveness without providing a major compensating improvement in utility. The complete PromptDLP configuration achieved the strongest security outcome, with 99.1% leakage prevention while maintaining approximately 98.6% utility preservation.



igure 3. Relationship between leakage-prevention efficiency and utility preservation across PromptDLP and comparative configurations.

Hierarchical clustering further revealed distinct similarities among the evaluated adversarial transformations (Figure 4). Direct exposure and whitespace or delimiter manipulation clustered relatively closely, reflecting their

Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ Research Article

similar and comparatively high detection profiles. Unicode and long-context transformations also showed relatively similar behaviour. In contrast, fragmented, encoded, and multi-source attacks occurred at greater linkage distances, indicating more distinct and challenging leakage patterns. Indirect prompt injection showed an intermediate relationship with the more complex attack group. The clustering pattern in Figure 4 therefore suggests that adversarial transformations can be broadly separated into simpler representation-level modifications and more complex distributed or structurally transformed leakage attempts.

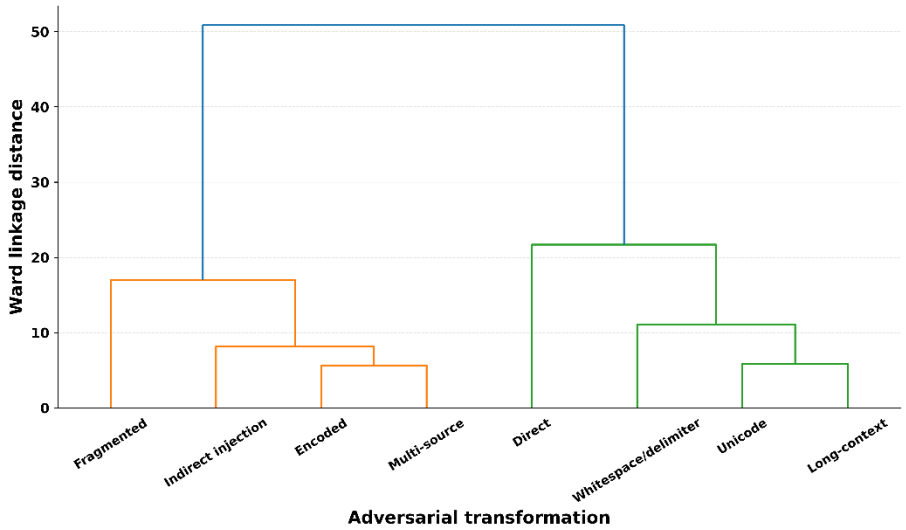


Figure 4. Hierarchical clustering of adversarial transformations based on leakage-prevention profiles across protection approaches.

DISCUSSION

Effectiveness of deterministic pre-inference control

The present findings demonstrate that deterministic inspection at the inference boundary can substantially reduce the probability of transmitting sensitive information to an LLM endpoint. PromptDLP achieved 98.7% precision, 99.3% recall, and 99.1% request-level leakage prevention, markedly exceeding both conventional regex filtering and model-assisted detection (Table 1). The particularly high recall is important in data-loss-prevention applications because false negatives represent direct opportunities for secret or PII disclosure (Gundaboina et al., 2023). Regex-based filtering remained useful for clearly structured identifiers but performed less effectively when sensitive values departed from familiar formats. Model-assisted detection improved coverage but introduced greater false-positive rates and less reproducible decisions (Wang et al., 2024). PromptDLP therefore appears to benefit from combining multiple deterministic mechanisms rather than depending solely on either syntactic pattern matching or probabilistic model judgment. Its 100% decision consistency further supports its suitability for environments where identical security events are expected to produce reproducible and auditable enforcement outcomes.

Protection across heterogeneous sensitive information

The consistently high performance across different sensitive-data categories indicates that the proposed framework was not dependent on a narrow class of secrets (Table 2). Recall remained above 98.8% for PII, financial identifiers, access tokens, credentials, cryptographic material, and organization-specific identifiers. API keys and access tokens showed the strongest performance, likely because such secrets commonly contain recognizable prefixes, length constraints, or high-entropy characteristics that can be validated deterministically (Rahmatulloh et al., 2019). The slightly lower performance for cryptographic fragments and organization-specific patterns highlights a more difficult challenge. These values can be structurally variable or resemble legitimate application data, increasing the possibility of either missed detections or false alarms. This suggests that practical enterprise deployment should combine general-purpose detectors with organization-specific policies and continuously maintained secret schemas rather than assuming that a universal rule set will adequately protect every data environment (Ge, 2022).

Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ Research Article

Robustness against adversarial evasion

The adversarial evaluation provides stronger evidence of PromptDLP's security value than performance on plaintext sensitive data alone. Leakage prevention remained above 97% under Unicode manipulation, encoding, fragmentation, indirect prompt injection, long-context placement, and multi-source disclosure (Figure 1). These transformations are particularly relevant to agentic systems because sensitive information may be assembled dynamically from several tools rather than occurring as a single recognizable string. The cluster analysis further showed that fragmented, encoded, and multi-source attacks represented relatively distinct and challenging attack patterns (Figure 4). Their separation from direct and delimiter-based attacks indicates that simple syntactic detection is unlikely to provide sufficient protection in complex agent workflows (Taylor et al., 2020). PromptDLP's strong performance in these conditions supports the importance of canonicalizing the complete outbound payload before evaluating egress policy.

Security and agent utility trade-off

An important concern with aggressive DLP systems is that increased protection can reduce the usefulness of legitimate applications. The results indicate that PromptDLP largely avoided this problem. The full configuration prevented 99.1% of sensitive leakage while preserving approximately 98.6% of baseline task utility (Figure 3). This balance is important because indiscriminate blocking could trivially achieve strong confidentiality while rendering an agent operationally ineffective (Chandrasekaran et al., 2021). The observed performance suggests that redaction, masking, and deterministic placeholder substitution can preserve enough contextual structure for many legitimate tasks without exposing the original protected values (Albanese et al., 2023). Nevertheless, the remaining utility loss indicates that certain workflows genuinely depend on sensitive information. In such situations, destination authorization and workflow-specific policy should determine whether processing is permitted rather than weakening the detector itself.

Computational feasibility and deployment implications

Security mechanisms inserted into every inference request must also introduce sufficiently low overhead for interactive applications. PromptDLP showed predictable scaling, with median inspection latency increasing from 3.2 ms for 1,000-token payloads to 26.3 ms for 32,000-token payloads (Figure 2). This approximately linear trend suggests that inspection cost is largely proportional to payload size and does not exhibit severe computational escalation within the evaluated context range (Máthé & Buşoniu, 2015). Such overhead is small relative to the typical latency associated with remote LLM inference and multi-step agent execution. The combination of high leakage prevention, deterministic decisions, preserved utility, and modest latency therefore suggests that pre-inference egress control can function as a practical additional security boundary (Yin et al., 2023). Rather than replacing prompt-injection defenses, access control, tool permissions, or output monitoring, PromptDLP can complement these mechanisms by ensuring that prohibited secrets and PII are intercepted before they are exposed to the model itself.

LIMITATIONS AND FUTURE RESEARCH DIRECTIONS

Despite the strong performance of PromptDLP, several limitations should be acknowledged. First, the evaluation was conducted using a controlled synthetic benchmark rather than production-scale datasets containing naturally occurring PII and operational secrets. Although synthetic data allowed precise ground-truth construction and avoided privacy risks, real enterprise environments may contain more ambiguous identifiers, proprietary formats, multilingual content, nested structured data, and organization-specific secrets that are difficult to capture through predefined rules. Second, the threat model assumed that attackers could manipulate prompts and untrusted content but could not compromise the PromptDLP policy engine or bypass the protected inference interface. Consequently, host-level compromise, insider threats, side-channel leakage, covert encoding techniques, and attacks against the DLP infrastructure itself were not evaluated. The study also examined a limited set of agent workflows and adversarial transformations, and the observed latency and utility preservation may vary across deployment environments, hardware configurations, model providers, context sizes, and high-throughput multi-agent systems. Furthermore, deterministic detection may remain vulnerable to previously unseen representations

of sensitive information, particularly when secrets are heavily transformed, semantically implied rather than explicitly stated, or distributed across multiple requests.

Future research should therefore validate PromptDLP using larger and more heterogeneous enterprise-scale workloads, including multilingual PII, proprietary identifiers, structured database records, source-code repositories, multimodal inputs, and dynamically generated secrets. Evaluation should also be extended to multi-agent architectures in which information passes through several agents, tools, memory stores, and inference endpoints, creating more complex egress pathways. Particular attention should be given to advanced evasion strategies such as cross-request secret reconstruction, semantic obfuscation, nested encodings, partial token leakage, and coordinated multi-source exfiltration. Future implementations could investigate hybrid architectures in which deterministic rules remain responsible for final enforcement while auxiliary machine-learning or semantic models identify previously unknown sensitive patterns for subsequent policy review. Additional work should also examine adaptive organization-specific policies, hardware-accelerated inspection, streaming prompt analysis, and formal verification of policy enforcement. Finally, longitudinal deployment studies in operational environments are needed to assess policy-maintenance burden, false-positive costs, developer acceptance, auditability, scalability, and the ability of PromptDLP to satisfy evolving privacy and regulatory requirements without materially reducing the usefulness of LLM-agent systems.

CONCLUSION

PromptDLP demonstrates that deterministic pre-inference egress control can provide an effective security boundary for preventing secret and personally identifiable information leakage in LLM-agent workflows. By inspecting and enforcing policy on the complete model-bound payload before inference, the framework achieved 98.7% precision, 99.3% recall, 99.0% F1-score, and 99.1% leakage prevention while maintaining 100% decision consistency. Its effectiveness remained above 97% across diverse adversarial transformations, including encoding, Unicode manipulation, fragmentation, indirect prompt injection, and multi-source disclosure, indicating resilience beyond conventional pattern-based filtering. Importantly, this security improvement was achieved while preserving approximately 98.6% of agent task utility and introducing only modest, predictable processing overhead, with median latency remaining below 30 ms even for 32,000-token payloads. These findings show that sensitive-data protection need not depend on the probabilistic behavior or instruction-following capability of the underlying LLM. Instead, deterministic enforcement at the inference boundary can provide reproducible, auditable, and model-independent control over sensitive-data egress. Although further validation using production-scale, multilingual, multimodal, and multi-agent environments is necessary, PromptDLP provides a promising architectural foundation for strengthening confidentiality, privacy, and governance in increasingly autonomous LLM-agent systems.

REFERENCES

[1] Acher, M., Duarte, J. G., & Jézéquel, J. M. (2023, August). On programming variability with large language model-based assistant. In Proceedings of the 27th ACM international systems and software product line conference-volume A (pp. 8-14).

[2] Albanese, F., Ciolek, D., & D'Ippolito, N. (2023). Text sanitization beyond specific domains: Zero-shot redaction & substitution with large language models. arXiv preprint arXiv:2311.10785.

[3] Chandrasekaran, V., Jia, H., Thudi, A., Travers, A., Yaghini, M., & Papernot, N. (2021). SoK: Machine learning governance. arXiv preprint arXiv:2109.10870.

[4] Chang, W. Y., Abu-Amara, H., & Sanford, J. F. (2011). Challenges of Enterprise Cloud Services1. In Transforming Enterprise Cloud Services (pp. 133-187). Springer, Dordrecht.

[5] Fasha, M., Rub, F. A., Matar, N., Sowan, B., Al Khaldy, M., & Barham, H. (2024, February). Mitigating the owasp top 10 for large language models applications using intelligent agents. In 2024 2nd International Conference on Cyber Resilience (ICCR) (pp. 1-9). IEEE.

[6] Gan, Y., Yang, Y., Ma, Z., He, P., Zeng, R., Wang, Y., ... & Ji, S. (2024). Navigating the risks: A survey of security, privacy, and ethics threats in llm-based agents. arXiv preprint arXiv:2411.09523.

[7] Ge, Z. (2022). Knowledge graphs and its applications in finance. The future and fintech: ABCDI and Beyond, 155-214.

[8] Giffin, D., Levy, A., Stefan, D., Terei, D., Mazières, D., Mitchell, J., & Russo, A. (2017). Hails: Protecting data privacy in untrusted web applications. Journal of Computer Security, 25(4-5), 427-461.

[9] Gundaboina, A. (2023). Data Loss Prevention in Healthcare: Advanced Strategies for Protecting PHI in Cloud Environments. Journal of Artificial Intelligence, Machine Learning & Data Science, 1(2), 3045-3051.

[10] Máthé, K., & Buşoniu, L. (2015). Vision and control for UAVs: A survey of general methods and of inexpensive platforms for infrastructure inspection. Sensors, 15(7), 14887-14916.

[11] Mayer, C. W., Ludwig, S., & Brandt, S. (2023). Prompt text classifications with transformer models! An exemplary introduction to prompt-based learning with large language models. Journal of Research on Technology in Education, 55(1), 125-141.

[12] Rahmatulloh, A., Gunawan, R., & Nursuwars, F. M. S. (2019, July). Performance comparison of signed algorithms on JSON Web Token. In IOP Conference Series: Materials Science and Engineering (Vol. 550, No. 1, p. 012023). IOP Publishing.

[13] Singh, J., Cobbe, J., & Norval, C. (2018). Decision provenance: Harnessing data flow for accountable systems. IEEE Access, 7, 6562-6574.

[14] Taylor, M., Vaidya, R., Davidson, D., De Carli, L., & Rastogi, V. (2020, November). Defending against package typosquatting. In International conference on network and system security (pp. 112-131). Cham: Springer International Publishing.

[15] Wang, S. J., Yao, J., Pei, K., Takahashi, H., & Yang, J. (2024). Detecting buggy contracts via smart testing. arXiv preprint arXiv:2409.04597.

[16] Yang, K., Liu, J., Wu, J., Yang, C., Fung, Y. R., Li, S., ... & Zhai, C. (2024). If llm is the wizard, then code is the wand: A survey on how code empowers large language models to serve as intelligent agents. arXiv preprint arXiv:2401.00812.

[17] Yin, J., Martin, J. P., Venkataramani, A., Alvisi, L., & Dahlin, M. (2003, October). Separating agreement from execution for byzantine fault tolerant services. In Proceedings of the nineteenth ACM symposium on Operating systems principles (pp. 253-267).

[18] Zhang, C., He, S., Qian, J., Li, B., Li, L., Qin, S., ... & Zhang, Q. (2024). Large language model-brained gui agents: A survey. arXiv preprint arXiv:2411.18279.