Toward an Autonomous Purple Teaming Framework for Security and Safety in Large Language Models

Leo Hyun Park
Graduate School of Information
Yonsei University
Seoul, South Korea
dofi@yonsei.ac.kr

YoonSik Kim
Graduate School of Information
Yonsei University
Seoul, South Korea
angelo1004@yonsei.ac.kr

Eunbi Hwang
Graduate School of Information
Yonsei University
Seoul, South Korea
ebhwang95@yonsei.ac.kr

Sangsoo Han
Graduate School of Information
Yonsei University
Seoul, South Korea
2023521505@yonsei.ac.kr

Hyoungshick Kim
Computer Science and Engineering
Sungkyunkwan University
Suwon, South Korea
hyoung@skku.edu

Taekyoung Kwon*
Graduate School of Information
Yonsei University
Seoul, South Korea
taekyoung@yonsei.ac.kr

***Abstract*—Large Language Models (LLMs) have rapidly advanced in reasoning capability and accessibility, driving their deployment across diverse applications. Yet this progress has also widened the surface for safety and security vulnerabilities. Adversaries can exploit prompt diversity, dialog memory, or multimodal inputs to induce unsafe or confidential outputs, while continual fine-tuning and third-party integration render static assurance infeasible. This paper introduces our ongoing national R&D project on developing the *AutoPT Framework*—an *Autonomous Purple Teaming* architecture that extends the collaborative principles of *purple teaming* toward self-adaptive, continuously verifiable LLM assurance. AutoPT unifies autonomous adversarial exploration and adaptive defensive reinforcement through two co-evolving agents. The red module, *AutoPT-Red*, employs coverage-guided fuzzing and internal measurement metrics to autonomously uncover vulnerabilities. The blue module, *AutoPT-Blue*, performs self-healing adaptation by updating guardrails and detecting integrity or confidentiality violations using embedding-based feedback. Preliminary case studies on jailbreak fuzzing and backdoor-poisoning defense validate the feasibility of this closed-loop, self-adapting architecture. As part of a broader national initiative, this work lays the conceptual and technical foundation for transitioning industrial purple teaming into a fully autonomous, scalable, and measurable assurance paradigm for generative AI systems.**

***Index Terms*—AI security, Large Language Models, Autonomous Purple Teaming, AI Security and Assurance, Jailbreak Detection, Backdoor Integrity Defense, Adaptive Guardrails**

I. Introduction

Large Language Models (LLMs) have emerged as a foundational component of modern AI ecosystems, powering conversational agents, code assistants, and multimodal reasoning systems across diverse domains. Their expanding reasoning capabilities and accessibility have accelerated widespread deployment in both academic and industrial settings. However, this progress has also broadened the attack surface for safety and security vulnerabilities. Adversaries can exploit prompt diversity, multi-turn dialog histories, or multimodal contexts to induce unsafe or confidential outputs. Moreover, continual fine-tuning, model updates, and third-party integrations frequently shift security boundaries, rendering static or one-time assurance approaches ineffective.

Recent academic research has proposed various frameworks for probing and safeguarding LLMs. Red-teaming methods such as GPTFuzzer [1], AutoDAN [2], AutoDAN-Turbo [3], and Papillon [4] automate the discovery of jailbreaks and unsafe behaviors. In the industrial domain, open-source frameworks such as Garak [5], PyRIT [6], and Giskard [7] provide scalable pipelines for generative AI auditing and risk evaluation, while defensive systems such as Llama Guard [8] and NeMo Guardrails [9] focus on input–output filtering and content moderation. Most recently, Meta's *Purple Llama* initiative [10] has formalized the concept of collaborative *purple teaming* for open-foundation models, emphasizing shared safety benchmarks, community red-teaming, and standardized guardrails. Note that these efforts still rely on human-orchestrated coordination and static evaluation pipelines.

This paper introduces our ongoing national R&D project, *Developing Techniques for Analyzing and Assessing Vulnerabilities, and Tools for Confidentiality Evaluation in Generative AI Models*. The project aims to establish a comprehensive foundation for vulnerability analysis and confidentiality assurance in generative AI systems, fostering trustworthy and self-adaptive AI security technologies. Within this initiative, we present the *AutoPT Framework*—an *Autonomous Purple Teaming* architecture that extends the principles of Purple Llama toward continuous, self-measuring, and self-healing security evaluation for LLMs.

AutoPT unifies autonomous adversarial exploration with

This work was supported by the Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. RS-2024-00439762, Developing Techniques for Analyzing and Assessing Vulnerabilities, and Tools for Confidentiality Evaluation in Generative AI Models).

*Corresponding author.

adaptive defensive reinforcement through two co-evolving agents. The red module, *AutoPT-Red*, employs coverage-guided fuzzing and internal measurement metrics to autonomously uncover vulnerabilities in LLMs. The blue module, *AutoPT-Blue*, performs self-healing adaptation by updating guardrails and detecting safety or confidentiality violations using embedding-based feedback. Together, these components form a closed feedback loop that enables self-measuring and continuously improving LLM assurance.

Preliminary case studies—focusing on jailbreak fuzzing and backdoor-poisoning defense—demonstrate the feasibility of this approach. As part of our ongoing project, this paper lays the conceptual and technical groundwork for evolving industrial purple-teaming initiatives such as Purple Llama into an autonomous, scalable, and resilient assurance framework for generative AI.

II. Background

LLM applications integrate foundation models with external systems to enable intelligent automation and decision support. Unlike traditional software that executes deterministic logic, LLMs generate outputs probabilistically based on contextual prompts. This nondeterminism, while central to their adaptability, also introduces attack surfaces that conventional security mechanisms were not designed to handle. Adversaries can manipulate model behavior through crafted prompts, poison external data sources in Retrieval-Augmented Generation (RAG), or exploit tool-use capabilities to trigger unintended actions [11]. As LLMs become increasingly agentic—capable of invoking APIs, accessing plugins, and interacting with private data—their operational boundaries blur, demanding new approaches to security assurance [12].

A. LLM-Specific Threats

The OWASP Top 10 for LLM Applications (2025) identifies unique risks in generative AI systems [13]. Prompt injection and excessive agency (LLM01, LLM06) allow adversaries to hijack model behavior or invoke unauthorized tools. Sensitive information disclosure and system prompt leakage (LLM02, LLM07) expose training data or internal logic. Data and model poisoning (LLM04) corrupt fine-tuning or retrieval datasets, while embedding vulnerabilities (LLM08) compromise vector-based retrieval. Finally, unbounded resource consumption (LLM10) can be exploited to exhaust computational capacity. Together, these threats reveal the fragility of current LLM ecosystems, where a single misaligned component can cascade into broader system compromise.

B. Threat Model and Existing Mitigations

We consider an enterprise LLM system equipped with RAG, tool invocation, and plugin extensions. Table I summarizes the assumed threat model, outlining key assets, adversary capabilities, and operational boundaries. Defenses today rely primarily on static guardrails—input sanitization, schema validation, rate limiting, and manual review. While effective against basic misuse, these controls lack adaptability and do not scale to dynamic, probabilistic systems. Their static nature limits continuous assurance, leaving organizations reactive rather than resilient.

TABLE I
Threat Model for Agentic LLM Applications.

Aspect	Description
Assets	Enterprise data in vector stores, system prompts, API credentials, and downstream services.
Adversaries	External attackers (prompt injection, resource abuse) and insiders (data exfiltration, poisoning).
Capabilities	Ability to submit arbitrary text inputs, influence indexed data sources, or exploit tool interfaces.
Constraints	Cannot modify model weights or infrastructure; network links are encrypted, but application-layer attacks remain viable.
Boundaries	User–Model–Tool/Plugin interfaces define trust separation in multi-tenant environments.

C. Need for Automated Evaluation

Existing frameworks [13], [14] classify threats but rarely automate their detection or mitigation. Adversarial testing remains ad hoc and expensive, providing incomplete coverage against evolving attack strategies. LLM applications thus require a continuous, automated, and feedback-driven security mechanism that jointly probes and strengthens models under real-world conditions. This necessity motivates our proposed AutoPT Framework, an automated purple-teaming system designed to evaluate and reinforce the security of generative AI applications through co-evolving red and blue team processes.

III. Our Framework

A. Overview

The proposed *AutoPT Framework* aims to establish an *autonomous purple-teaming* architecture capable of continuously assessing and reinforcing the security and safety of large language models (LLMs). It integrates two complementary and co-evolving modules—*AutoPT-Red* for adversarial discovery and *AutoPT-Blue* for defensive adaptation—within a closed-loop feedback system.

Unlike traditional automated pipelines, AutoPT pursues three dimensions of autonomy: (1) *self-exploration*, the ability to independently generate and optimize attack strategies from minimal or empty seeds; (2) *self-measurement*, using internal model signals to quantify behavioral vulnerabilities; and (3) *self-healing*, the capacity to adapt and reconfigure defenses based on discovered weaknesses. Through this triad, AutoPT evolves beyond static testing frameworks toward a self-improving security ecosystem.

As illustrated in Figure 1, the system operates as a cyclic process: AutoPT-Red autonomously probes vulnerabilities and produces embedding-based vulnerability descriptors. These embeddings are consumed by AutoPT-Blue, which retrains or refines its guardrails and safety classifiers. Updated defensive boundaries are then fed back into AutoPT-Red to guide its next iteration of exploration, forming a continuous co-adaptive cycle.

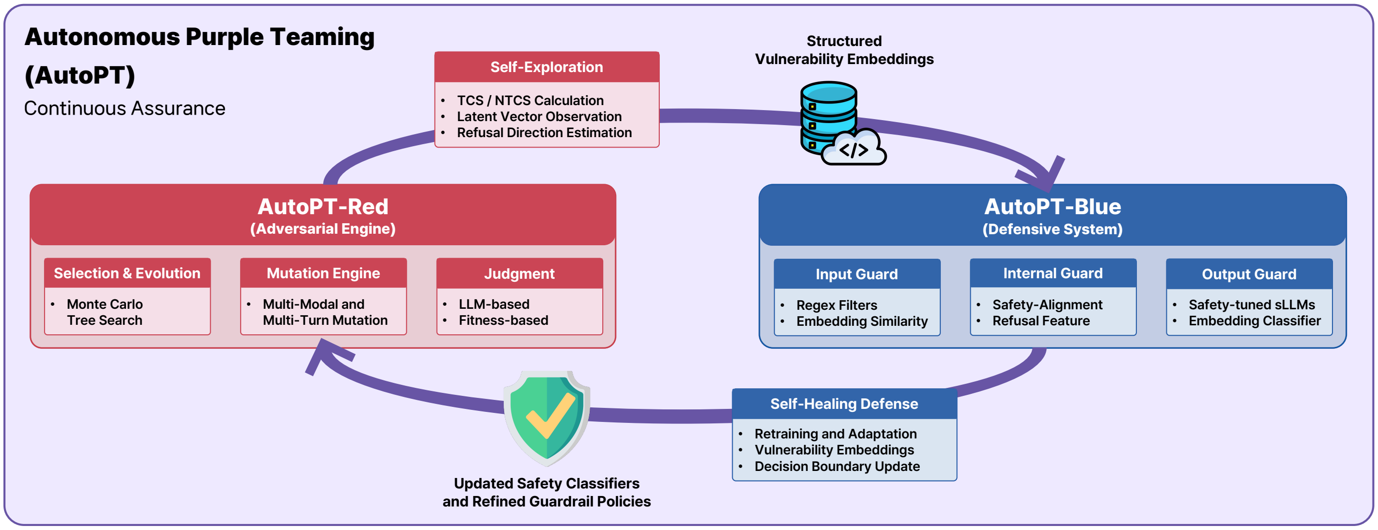

Fig. 1. The overview of our framework.

B. *AutoPT-Red: Autonomous Adversarial Exploration*

Objective. AutoPT-Red is designed as a *self-exploratory fuzzing agent* that autonomously uncovers jailbreak strategies and safety failures in LLMs. It extends the coverage-guided fuzzing paradigm by incorporating internal measurement feedback, enabling gradient-informed reasoning about model compliance and refusal behavior.

Architecture. The AutoPT-Red module comprises three functional layers:

1) *Selection and Evolution:* The selection process operates on a candidate prompt pool that may initially be empty or minimally seeded, prioritizing higher-scoring candidate prompts via a Monte Carlo Tree Search (MCTS)–based selector that balances exploration (novel behaviors) and exploitation (confirmed jailbreaks).
2) *Mutation Engine:* The selected candidate prompts are subsequently diversified through multimodal and multi-turn transformations, including text paraphrasing, context grafting, retrieval injection, and image conditioning. This mutation process can operate without predefined templates, as the agent autonomously synthesizes and refines prompts via self-rewriting and persona-role induction, inspired by AutoDAN-Turbo [3] and Papillon [4].
3) *Judgment:* Each mutated test case is then evaluated, and successful ones are added back to the AutoPT-Red prompt pool. The evaluation employs an LLM-based judgment model to estimate the attack success rate (ASR), while internal metrics such as log probability can also serve as fitness functions guiding this judgment.

Self-Exploration. AutoPT-Red autonomously explores the behavioral space of the target model through internal measurement feedback rather than external supervision. Each candidate prompt is internally assessed using model-intrinsic signals that capture shifts in compliance behavior and gradient sensitivity within the generation process. These measurements act as intrinsic fitness functions that guide the Monte Carlo Tree Search (MCTS)–based exploration toward behaviorally diverse and high-risk regions. By continuously interpreting and responding to these internal cues, AutoPT-Red adapts its search direction and progressively refines its strategy, enabling self-directed optimization without human intervention.

Output. The final output of AutoPT-Red is a structured set of *vulnerability embeddings* representing discovered weaknesses. Each embedding integrates multiple internal representation–based measurements—such as compliance-score estimation, latent-vector observation, and refusal-direction analysis—capturing contextual features (e.g., modality, turn index, risk type) and sensitivity patterns. These embeddings provide a compact yet semantically rich description of model vulnerabilities, serving as transferable inputs for downstream defensive adaptation.

C. *AutoPT-Blue: Autonomous Defensive Adaptation*

Objective. AutoPT-Blue serves as the defensive counterpart to AutoPT-Red, implementing continuous guardrail adaptation and self-healing defense updates. It autonomously ingests vulnerability embeddings, retrains classifiers, and refines response policies—transforming discovered weaknesses into strengthened safety boundaries.

Architecture. AutoPT-Blue is composed of four primary components:

1) *Input Guard:* It performs proactive filtering of incoming prompts through hybrid detection strategies that combine embedding similarity against known red vulnerabilities, lexical constraint analysis, and contextual anomaly scoring.
2) *Internal Guard:* Maintains the model's internal safety alignment by continuously auditing intermediate activations and refusal features, ensuring that the learned safety representations remain consistent against backdoor or

trojan-like perturbations. It correlates these patterns with vulnerability embeddings to assess integrity risk.

3) *Output Guard:* Evaluates generated responses using safety-tuned sub-LLMs (sLLMs) and embedding-based classifiers to detect unsafe or confidential outputs.

Self-Healing Defense. Upon receiving newly identified vulnerability embeddings from AutoPT-Red, AutoPT-Blue autonomously initiates a retraining and adaptation cycle to reinforce its guardrails. This process re-clusters vulnerability embeddings, samples representative cases, and incrementally fine-tunes defensive classifiers. During this phase, the system synthesizes feedback from all guards and applies low-rank adaptation or conditional layer steering to update decision boundaries automatically, effectively transforming red-team discoveries into blue-team immunization without human intervention.

Output. The outputs of AutoPT-Blue include (i) updated safety classifier weights, (ii) refined prompt and response policies, and (iii) metadata describing defensive evolution. These updates are shared back to AutoPT-Red through the embedding exchange interface, allowing the red module to reinitiate exploration under new safety boundaries.

D. Autonomous Co-evolution and Measurement

The strength of AutoPT lies in its self-measuring co-evolution cycle. The scoring metric serves as the quantitative backbone of AutoPT, providing a unified measurement paradigm that ensures interpretability and traceability across red–blue iterations. AutoPT-Red uses its internal fitness function to drive exploration, while AutoPT-Blue employs aggregated fitness trends to adjust its thresholds and retraining triggers. Through this loop, AutoPT progressively learns to anticipate, quantify, and mitigate vulnerabilities autonomously—laying the groundwork for a scalable, continuously adaptive LLM security ecosystem.

IV. Case Studies

This section presents two preliminary case studies demonstrating the feasibility of the proposed *AutoPT Framework*. The first illustrates the autonomous red-teaming capability (*AutoPT-Red*) using a coverage-guided fuzzing engine driven by the Token Control Score (TCS) [15]. The second highlights the autonomous blue-teaming capability (*AutoPT-Blue*) through embedding-based backdoor integrity violation detection and adaptive guardrail updates.

A. AutoPT-Red: Coverage-Guided Fuzzing for Jailbreak Discovery

To describe the red-side ongoing work of AutoPT, we review our prior work on Normalized TCS (NTCS) [15] within the coverage-guided fuzzing framework.

Scoring Metrics. TCS serves as a quantitative indicator that measures how effectively a jailbreak prompt steers a model from refusal toward compliance. Given logits $Z_P(t)$ and $Z_{P^\star}(t)$ for token t under the original malicious prompt P and the jailbreak-modified prompt $P^\star$, the score is defined as

$$\text{TCS} = \big(Z_P[r'] - Z_{P^\star}[r']\big) - \big(Z_P[c'] - Z_{P^\star}[c']\big), \quad (1)$$

where r' and c' denote the most probable refusal and compliance prefix tokens, respectively. A higher TCS indicates a stronger behavioral shift of the model's response toward compliance.

To enhance discriminability, the *Normalized TCS (NTCS)* [15] incorporates the gradient magnitude $\|\nabla_A \text{TCS}\|_2$ of the final self-attention layer:

$$\text{NTCS} = \frac{\text{TCS}}{\|\nabla_A \text{TCS}\|_2}. \quad (2)$$

NTCS provides a continuous fitness signal that captures both the degree of behavioral shift and the gradient sensitivity of the model. This measurement is expected to serve as the intrinsic fitness function for AutoPT-Red in the current stage, whereas more functions and methodologies will be explored in our ongoing study.

LLM Fuzzing. Building on this metric, our case study [15] implements coverage-guided fuzzing on top of LLM-Fuzzer [1]. Note that in our ongoing work of AutoPT, we furhter integrate self-exploratory fuzzing inspired by AutoDAN-Turbo [3] and Papillon [4].

Experiments on open-weight models including Llama-3-8B, Llama-2-7B, and Mistral-7B demonstrate that our case study achieves up to *95%* jailbreak success with approximately half the query budget of LLM-Fuzzer [1]. Moreover, generated templates exhibit high universality across diverse tasks and strong transferability to proprietary (closed black box) models such as GPT-4 and Claude 3. These results indicate that NTCS-driven, coverage-guided fuzzing enables effective and efficient discovery of novel jailbreak templates without human supervision, expecting and validating our red-side ongoing work of AutoPT.

B. Blue: Model Integrity Verification against Tampering

To demonstrate the defensive capability of our framework, we adopt a prefix-based fingerprinting method for detecting model tampering. This mechanism mitigates supply-chain risks such as unauthorized modification, malicious fine-tuning, or model substitution, forming the foundation of model-level integrity assurance within the AutoPT-Blue module.

Threat Model. We consider a black-box setting where a model owner releases M_{base} through untrusted channels. Adversaries may (1) replace it with an altered model M_ϕ, (2) modify parameters via fine-tuning or Trojan insertion, or (3) redistribute a derivative version. Tampering is assumed to cause detectable behavioral deviations, and attackers do not know the private fingerprint set $\mathcal{D}_{\text{fp}}$. Verification relies solely on query–response pairs.

Fingerprint-Based Verification. A lightweight prefix adapter M_{prefix} is attached to the public base model M_{base} to embed a private fingerprint mapping. The verifier maintains a secret dataset $\mathcal{D}_{\text{fp}} = \{(x_{\text{fp}}^{(i)}, y_{\text{fp}}^{(i)})\}_{i=1}^{N}$ consisting of cryptographically random fingerprint pairs indistinguishable from normal queries. During training, M_{prefix} is optimized while keeping M_{base} frozen to minimize

$$\mathcal{L}_{\text{fp}} = \frac{1}{N}\sum_{i=1}^{N} \ell(M_{\text{prefix}} \circ M_{\text{base}}(x_{\text{fp}}^{(i)}), y_{\text{fp}}^{(i)}),$$

Thereby binding the fingerprint to the model's original behavior. For verification, the deployed model is queried with x_{fp}, and integrity is confirmed if

$$\frac{1}{N}\sum_{i=1}^{N} \nVdash \Big[d(M_{\text{prefix}} \circ M_{\text{base}}(x_{\text{fp}}^{(i)}), y_{\text{fp}}^{(i)}) < \epsilon\Big] \geq \theta,$$

where $d(\cdot,\cdot)$ is a distance metric, ϵ a tolerance, and θ the acceptance threshold.

Security and Limitations. The method provides three guarantees: (1) tamper sensitivity, since any modification to M_{base} breaks the fingerprint mapping; (2) unforgeability, because $\mathcal{D}_{\text{fp}}$ and M_{prefix} remain private; and (3) efficiency, as M_{prefix} contains fewer than 0.1% of total parameters and incurs negligible overhead. Limitations include secure storage of private components and careful tuning of ϵ to balance false positives and negatives.

In summary, this fingerprint-based mechanism enables lightweight and verifiable model integrity checks under a fully black-box threat model, offering practical protection against supply-chain tampering.

V. Discussion

Toward Autonomous Purple Teaming in Practice. The AutoPT Framework represents an initial realization of the national R&D project *"Developing Techniques for Analyzing and Assessing Vulnerabilities, and Tools for Confidentiality Evaluation in Generative AI Models."* It advances the emerging paradigm of *purple teaming*—as formalized by initiatives such as Meta's Purple Llama—toward a self-directed, continuously evolving security ecosystem for generative AI. By integrating measurement-driven adversarial exploration (*AutoPT-Red*) and embedding-driven self-healing defense (*AutoPT-Blue*), AutoPT establishes a closed feedback loop in which both offensive and defensive intelligence adapt dynamically to new model behaviors. Unlike static audits or manually orchestrated red–blue testing, AutoPT demonstrates that LLM security validation can progress toward *self-exploration, self-measurement, and self-adaptation*—three foundational pillars of autonomy.

The key innovation lies in its *measurement layer*, where internal model signals such as the Token Control Score (TCS) provide interpretable, quantitative assessments of vulnerability severity. This metric-driven architecture forms a shared substrate between red and blue agents, ensuring that attack discovery and defense refinement are guided by a coherent objective function rather than discrete, human-defined rules.

Scalability and Multi-Agent Evolution. While the current implementation focuses on single-loop experimentation, the AutoPT architecture generalizes naturally to multi-agent ecosystems. An *AutoPT Swarm* could consist of distributed red and blue agents operating asynchronously across heterogeneous generative models (e.g., LLMs, diffusion models, and multimodal LMMs), exchanging embeddings and risk summaries through a shared measurement space. Such distributed co-evolution would enable:

- Large-scale vulnerability coverage across diverse architectures and deployment settings;
- Continuous benchmarking under realistic, multi-tenant conditions; and
- Dynamic policy alignment across collaborative or federated AI systems.

Future extensions may incorporate reinforcement-based scheduling or self-play dynamics to accelerate convergence toward stable, resilient safety boundaries.

Governance and Standardization Outlook. As LLMs increasingly underpin critical digital infrastructure, autonomous assurance frameworks like AutoPT will be essential for transparent and auditable AI governance. By generating interpretable and reproducible measurements of model risk, AutoPT can support standardized metrics for safety evaluation, alignment testing, and regulatory compliance. The TCS-based quantification of behavioral shift provides a practical bridge between research prototypes and industrial audit pipelines, complementing ongoing initiatives such as Purple Llama in establishing open, community-driven benchmarks for trustworthy AI. Its continuous measurement cycle can be embedded into larger enterprise workflows—enabling security dashboards, automated red–blue testing as a service, and self-reporting safety indicators for oversight bodies.

VI. Conclusion

This paper presented our ongoing work on the *AutoPT Framework*, an *Autonomous Purple Teaming* architecture developed under the national project *"Developing Techniques for Analyzing and Assessing Vulnerabilities, and Tools for Confidentiality Evaluation in Generative AI Models."* AutoPT embodies the project's vision of evolving traditional, human-centered purple teaming into an autonomous, self-adaptive process. Through its two co-evolving modules—*AutoPT-Red* and *AutoPT-Blue*—the framework operationalizes self-exploratory fuzzing, self-measuring vulnerability quantification, and self-healing defense adaptation.

Preliminary case studies on jailbreak fuzzing and backdoor-poisoning detection validate the feasibility of this approach, demonstrating improved efficiency, adaptability, and interpretability compared to existing automated pipelines. As part of an ongoing national initiative, AutoPT serves as both a conceptual and technical foundation for bridging academic research and industrial practice in AI assurance. It extends the principles of collaborative purple teaming, as exemplified by Purple Llama, toward full autonomy—where large language models continuously test, explain, and fortify themselves.

Future work will more concretize and extend AutoPT toward multi-agent autonomy, multimodal and federated risk assessment, and standardized measurement frameworks that enable scalable, verifiable, and resilient AI security in the era of generative intelligence.

REFERENCES

[1] J. Yu, X. Lin, Z. Yu, and X. Xing, "Llm-fuzzer: Scaling assessment of large language model jailbreaks," in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 4657–4674.

[2] X. Liu, N. Xu, M. Chen, and C. Xiao, "Autodan: Generating stealthy jailbreak prompts on aligned large language models," in *The Twelfth International Conference on Learning Representations(ICLR)*, 2024.

[3] X. Liu, P. Li, E. Suh, Y. Vorobeychik, Z. Mao, S. Jha, P. McDaniel, H. Sun, B. Li, and C. Xiao, "Autodan-turbo: A lifelong agent for strategy self-exploration to jailbreak llms," in *The Twelfth International Conference on Learning Representations (ICLR)*, 2025.

[4] X. Gong, M. Li, Y. Zhang, F. Ran, C. Chen, Y. Chen, Q. Wang, and K.-Y. Lam, "Papillon: Efficient and stealthy fuzz testing-powered jailbreaks for llms," in *34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 2401–2420.

[5] L. Derczynski, E. Galinkin, J. Martin, S. Majumdar, and N. Inie, "garak: A framework for security probing large language models," *arXiv preprint arXiv:2406.11036*, 2024.

[6] G. D. L. Munoz, A. J. Minnich, R. Lutz, R. Lundeen, R. S. R. Dheekonda, N. Chikanov, B.-E. Jagdagdorj, M. Pouliot, S. Chawla, W. Maxwell *et al.*, "Pyrit: A framework for security risk identification and red teaming in generative ai system," *arXiv preprint arXiv:2410.02828*, 2024.

[7] G. A. R. Team, "Giskard: Continuous red-teaming platform for llm agents," https://giskard.ai/, 2024.

[8] H. Inan, K. Upasani, J. Chi, R. Rungta, K. Iyer, Y. Mao, M. Tontchev, Q. Hu, B. Fuller, D. Testuggine, and M. Khabsa, "Llama guard: Llm-based input-output safeguard for human-ai conversations," *arXiv preprint arXiv:2312.06674*, 2023. [Online]. Available: https://arxiv.org/abs/2312.06674

[9] Y. Dong, R. Mu, G. Jin, Y. Qi, J. Hu, X. Zhao, J. Meng, W. Ruan, and X. Huang, "Building guardrails for large language models," *arXiv preprint arXiv:2402.01822*, 2024.

[10] Meta AI Research Team, "Purple llama: Open safety and trust tools for the open foundation model community," https://ai.meta.com/blog/purple-llama-open-safety-tools/, 2024, accessed: Oct 2025.

[11] B. C. Das, M. H. Amini, and Y. Wu, "Security and privacy challenges of large language models: A survey," *ACM Computing Surveys*, vol. 57, no. 6, pp. 1–39, 2025.

[12] R. Fang, R. Bindu, A. Gupta, and D. Kang, "Llm agents can autonomously exploit one-day vulnerabilities," *arXiv preprint arXiv:2404.08144*, 2024.

[13] OWASP, "Owasp top 10 for large language model applications," https://owasp.org/www-project-top-10-for-large-language-model-applications/, 2025.

[14] The MITRE Corporation, "MITRE ATLAS: Adversarial Threat Landscape for Artificial-Intelligence Systems," https://atlas.mitre.org, 2023.

[15] L. H. Park and T. Kwon, "Red-teaming llms with token control score: Efficient, universal, and transferable jailbreaks," in *Proceedings of the 28th International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2025)*, Gold Coast, Australia, 2025.