Proceedings of the
13th International Conference on Applied Informatics
Eger, Hungary, February 19–21, 2026
DOI: 10.17048/icai.2026.107
pp. 107–121

# LLM-Powered Automated Attacks

**Tamás Girászi, Natáli Papp, Norbert Oláh, Andrea Huszti**

Faculty of Informatics, University of Debrecen, Debrecen, Hungary
giraszi.tamas@inf.unideb.hu
pappnaty@gmail.com
olah.norbert@inf.unideb.hu
huszti.andrea@inf.unideb.hu

**Abstract.** The increasing integration of large language models (LLMs) into systems introduces new attack surfaces that extend beyond traditional software vulnerabilities. While LLMs are commonly protected by prompt-level security mechanisms, recent researches show that these controls can be bypassed through carefully crafted inputs. In this paper, we propose an interaction model based on a Generative Adversarial Network (GAN) conceptual analogy and a dual-LLM framework for systematically examining and testing the security boundaries of LLMs through malicious code generation. The framework consists of two LLMs that iteratively produce attack-oriented prompts, interpret and implement them. Experimental results demonstrate that the proposed approach can generate outputs that are similar to real-world attack patterns, such as SQL injection and cross-site scripting. This research highlights the importance of LLM security controls and emphasizes the need for proactive, automated evaluation methods to improve the robustness and governance of LLM-based systems.

*Keywords:* Large Language Models, Adversarial Prompting, LLMs, GAN

2020 *Mathematics Subject Classification:* Primary 60F15, 60G42, 60G50; Secondary 60F10

## 1. Introduction

Over the past decade, advances in artificial intelligence have had a huge impact on the IT domain. The rapid evolution of AI-driven methodologies, particularly machine learning (ML) and deep learning (DL), has significantly influenced not only economic and industrial processes but has also fundamentally reshaped the

cybersecurity landscape, affecting both defensive and adversarial capabilities. AI algorithms enable the efficient analysis of large-scale data and facilitate the identification of malicious patterns and anomalous behaviors with increasing accuracy and speed ([4, 17, 36, 41]).

Traditionally, every IT system is designed to meet fundamental security requirements. However, the growing complexity, operational speed, and adaptive characteristics of modern threats have introduced challenges that exceed the capabilities of classical, mainly reactive defense paradigms.[26] On the other hand, AI-based techniques have begun to appear on the offensive side, enabling adversaries to autonomously discover system vulnerabilities, optimize attack strategies, and generate highly targeted phishing and social engineering attacks ([10, 13, 14, 33]).

The offensive use of artificial intelligence introduces new dimensions to the nature of cyberattacks. Whereas earlier threat models relied mainly on automation and script-based techniques, modern attack frameworks increasingly incorporate self-learning and adaptive mechanisms capable of dynamically bypassing established defensive controls. LLMs, in particular, leverage advanced natural language processing capabilities to perform sophisticated conversational, text generation, and analytical tasks, thereby facilitating the large-scale automation of social engineering and disinformation activities. As a result, LLMs are increasingly employed to generate phishing emails, fraudulent communications, and persuasive textual content designed to manipulate or deceive users ([8, 13, 18, 45]).

The objective of this research is to develop an automated and proactive offensive framework that systematically exposes system vulnerabilities through the generation of adversarial attacks. The proposed approach is based on the interaction of two large language models, implemented as Generative Pre-trained Transformer (GPT) based systems and driven by distinct prompting strategies, within a Generative Adversarial Network (GAN) inspired architectural paradigm. In this configuration, one LLM functions as a generator that iteratively produces adversarial prompts until it induces the discriminator model to generate malicious content. This process enables the systematic evaluation of system-level security constraints. The proposed framework highlights the critical importance of robust AI security mechanisms and provides a foundation for the development of targeted defensive measures and validation protocols. Furthermore, it enables structured testing of LLM security controls, offering insight into the extent to which such models may be sensitive to manipulation or compromise.

## 2. Related Work

There are numerous types of cyber attacks that, driven by ongoing digitalization, affect multiple technological and organizational domains. The literature identifies several major categories of cyber attacks, the diversity and increasing complexity of these threats and illustrating the challenges faced by organizations and individuals operating in today's digitally interconnected environment ([26, 29, 32, 38]).

Many techniques are employed in various ways to support cyberattacks, including the automation of phishing campaigns and the generation of evasive malware variants, thereby increasing the effectiveness and scalability of adversarial activities ([8, 33]). Through the application of natural language processing and LLMs, adversaries are able to generate more convincing and seemingly legitimate messages, substantially enhancing the success rate of deception-based attacks. Such techniques not only improve the efficiency of attack execution but also considerably complicate their timely detection and mitigation ([13, 14]). LLMs must be equipped with robust defensive mechanisms capable of detecting and mitigating prompt injection attacks and consistently rejecting malicious prompts. However, such defenses are often relatively easy to circumvent, while their systematic evaluation and analysis remain inherently challenging. ([20, 42, 44])

Several recent studies explicitly investigate how large language models can be deliberately coerced into producing malicious or policy-violating outputs through prompt injection and jailbreaking techniques ([34, 42, 44]). Shen et al. [35] analyze thousands of real-world jailbreak prompts collected from online communities, with the explicit goal of identifying prompt patterns that successfully bypass built-in security mechanisms. Their work focuses on empirical characterization of existing attack prompts rather than generating new ones adaptively.

The security of LLMs extends beyond external threats such as prompt manipulation, data leakage, or malicious misuse (e.g., phishing or disinformation), and also encompasses internal risks arising from the autonomous operation of these models ([20, 34, 42]). These risks are especially in the case of autonomous LLM-based agents, where the objectives pursued by the model may diverge from user intent, strategic deceptive behaviors may emerge, or latent goal structures may persist despite extensive security and alignment training ([15]). Among these, it includes not only goal misalignment but also the potential emergence of hidden objectives, strategic deception (scheming), self-preserving behaviors, and the persistence of these undesirable properties even under current security training paradigms ([15, 28, 35]).

Yu et al. [43] propose GPTFUZZER, a black-box red-teaming framework that adapts the concept of software fuzzing to the security analysis of LLMs through the use of an initial jailbreak template corpus and evolutionary mutation operators. While their approach demonstrates strong capability in generating diverse and robust adversarial prompts via automated template exploration, its effectiveness heavily depends on the quality of the initial seeds and places less emphasis on fine-grained, context-specific semantic interaction between the attacker and the target model.

Mehrotra et al. [25] propose the TAP (Tree of Attacks with Pruning) method, an automated framework that combines tree-based search with output-driven pruning to systematically jailbreak black-box LLMs. While their approach demonstrates that structured reduction of the search space can significantly improve attack efficiency and reduce query costs, its effectiveness critically depends on the evaluator model's ability to accurately predict the target model's safety responses to

partially refined prompts. Additionally, the approach may exhibit limited generalization across models with different alignment strategies, incur non-trivial computational overhead due to tree expansion, and risk converging to locally optimal attack strategies, thereby overlooking more diverse jailbreak trajectories.

Chao et al. [5] propose the PAIR (Prompt Automatic Iterative Refinement) method, an automated framework that iteratively refines semantically grounded adversarial prompts against black-box LLMs through an "attacker–judge–target" loop. While their approach demonstrates the remarkable effectiveness of social engineering–based jailbreaking with a minimal number of queries, it requires the use of an additional LLM as part of the attack pipeline, which introduces extra computational cost and potential monetary overhead.

## 3. Our Contribution

This work presents a comprehensive, proactive-offensive, LLM-assisted security evaluation framework for the systematic analysis of prompt-level vulnerabilities in large language models. Its primary contributions are the design, implementation, and empirical evaluation of a novel GAN-inspired dual-LLM adversarial prompting architecture that models prompt manipulation as an iterative, self-supervised process, enabling continuous exploration of prompt-based attacks without any parameter modification or fine-tuning. Instead of optimizing a single attack objective or replaying known jailbreak patterns, the framework captures how adversarial prompts evolve through sustained interaction between a generator and a discriminator, thereby supporting the systematic study of vulnerability emergence and offering deeper insight into the adaptive dynamics that can drive malicious model behavior. Figure 1 illustrates the proposed architecture, and the resulting generated prompts can be leveraged to inform and develop more effective LLM defense mechanisms.

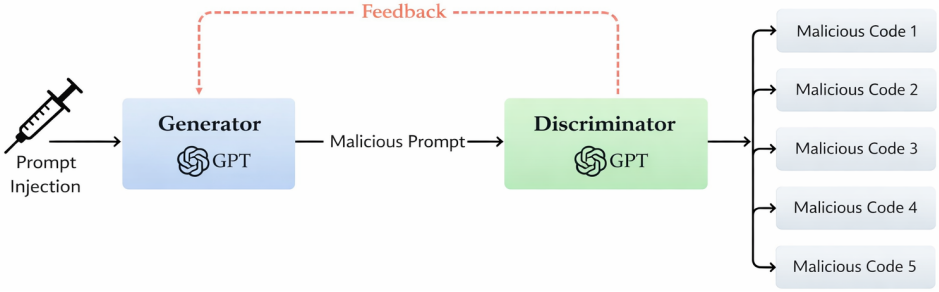


**Figure 1.** Proposed system structure.

Unlike conventional robustness evaluations that rely on static test sets or supervised fine-tuning, the proposed framework operates entirely at the input level. It leverages the interaction between two large language models in an adversarial

but cooperative configuration to expose failure modes arising from linguistic reformulation, contextual reframing, and indirect instruction. This approach allows us to adapt our AI-based system without teaching it ([12, 22]).

Furthermore, the proposed framework demonstrates that the use of LLMs in this manner can substantially amplify the capabilities of smaller, less resourced, or knowledge-constrained adversarial groups, potentially increasing their overall threat level.

In contrast, our work focuses on modelling adversarial prompt generation as a dual-LLM interaction process in which one model continuously reformulates attack prompts while the other model provides response-based feedback for subsequent finetuning. The objective is therefore not only to maximize attack success, but also to study how prompt-level vulnerabilities emerge through iterative linguistic reframing, contextual manipulation, and feedback-driven reformulation.

The proposed architecture does not implement gradient-based adversarial training, discriminator loss optimization, parameter updates, or end-to-end neural network training. Instead, the analogy refers to the functional separation between a prompt-generating component and an evaluating/responding component, whose interaction resembles an adversarial feedback loop. This distinction is important because the framework operates entirely at inference time and modifies only the textual input, not the model parameters.

Accordingly, the contribution of this paper is a proof-of-concept framework for proactive LLM security evaluation. Its purpose is to support systematic exploration of prompt-level failure modes and to provide empirical insight into how adversarial prompts can evolve through repeated reformulation.

The security concern examined in this work is demonstrated through malicious code generation. This work is not to evaluate the quality, completeness, or success rate of the generated code, but to show that the tested LLMs can be forced to produce outputs containing malicious attack patterns despite built-in safety restrictions. This is a security concern because the proposed framework was able to bypass prompt-level defenses through reformulation and contextual framing. Therefore, the generated outputs are treated as evidence of a security-relevant failure mode not because they are necessarily fully executable or optimized, but because they contain misuse-relevant logic that should normally be restricted.

## 3.1. Iterative Adversarial Prompting Process

Rather than learning a probability distribution through gradient-based optimization, the framework iteratively refines adversarial prompts and evaluates model responses through a closed feedback loop ([2, 42]).

This occurs for two reasons. First, such models are trained using extensive computational resources and large-scale datasets, the replication of this is both technically complex and economically costly. Second, access to these capabilities is available at a relatively low marginal cost, enabling users to leverage the results of large-scale training with minimal financial investment. As a consequence, adversarial actors can significantly reduce the time, effort, and resources required to

develop sophisticated attack capabilities.

The architecture consists of two functionally complementary components:

- **Generator:** The generator is responsible for constructing and iteratively refining adversarial prompt candidates. Its objective is to explore the space of potentially unsafe or policy-challenging instructions by producing linguistically varied, contextually framed prompts associated with specific attack scenarios (e.g., SQL injection, cross-site scripting, phishing logic).
- **Discriminator:** The discriminator evaluates the generator's outputs by responding to the generated prompts using its built-in security mechanisms and alignment constraints. The discriminator's responses implicitly contain whether a given prompt successfully elicits security-relevant behavior, thereby functioning as a risk assessor and feedback provider.

Although the generator and discriminator follow opposing objectives, they operate cooperatively by forming a continuous feedback loop. This interaction realizes a form of adversarial learning at the prompt level, where the generator's search strategy is shaped by the discriminator's responses rather than by explicit labels or gradients.

A central idea of this work is the formulation of an iterative, self-supervised adversarial prompting process. In each iteration, the generator produces a candidate prompt, which is then executed by the discriminator. Based solely on the discriminator's response, the generator adapts its subsequent prompt-generation strategy.

The iterative process follows four conceptual stages:

1. **Attack Goal Definition:** The generator identifies a high-level attack objective or threat category, such as injection-based attacks or social engineering tactics.
2. **Prompt Synthesis:** A candidate prompt is generated using predefined templates and contextual modifiers to produce the selected attack objective.
3. **Evaluation and Feedback:** The discriminator processes the prompt and generates a response.
4. **Refinement:** The generator refines the prompt formulation based on the discriminator's feedback and initiates the next iteration if needed.

When designing generator prompts, our goal is to ensure that malicious intent is not explicitly revealed when generating offensive prompts. We use implicit prompt injection techniques at the generator prompt level to hide the offensive nature of the generated prompt and maintain the appearance of a harmless context in two different ways. One approach is to modify the wording, focusing exclusively on lexical-level changes. The other approach is reframing, which modifies not only the words but also the meaning of the entire context in order to make the prompt appear harmless.

The discriminator component executes the prompts generated by the generator, which decides their acceptability based on its security. If the evaluation is unsuccessful, program code is generated, while if it is successful, the system starts a new generation-evaluation cycle to jailbreak the discriminator.

This mechanism is intentionally analogous to the iterative competition observed in classical GAN training, with the crucial distinction that no model parameter updates or training process are performed. All adaptation occurs at the prompt level, demonstrating that model behavior can be materially influenced through linguistic manipulation alone. The process shows how security mechanisms based on static alignment or rule-based filtering can be progressively weakened through contextual and semantic variation.

By generation adversarial prompt as a controlled evaluation process, the framework provides a principled methodology for studying how and why large language models may fail under adversarial linguistic pressure ([28, 35, 44]).

# 4. Preliminaries

## 4.1. Large Language Models (LLMs)

Large Language Models (LLMs) are autoregressive probabilistic models trained to predict the next token in a sequence given its preceding context ([20, 42]). For an input token sequence $x_{1:n}$, an LLM parameterized by $\theta$ models the conditional distribution

$$p_\theta(x_{1:n}) = \prod_{t=1}^{n} p_\theta(x_t \mid x_{<t}),$$

and generates text by iteratively sampling or decoding tokens from this distribution until a stopping criterion is met.

Modern LLMs are predominantly based on the Transformer architecture, which uses multi-head self-attention to compute context-aware token representations. This mechanism enables the model to capture long-range dependencies and to adapt its outputs based on subtle variations in prompt structure and context ([39]).

LLMs are typically pretrained using a next-token prediction objective on large-scale text corpora, allowing them to acquire broad linguistic and semantic knowledge[42]. Subsequent instruction tuning and alignment techniques, such as reinforcement learning from human feedback, are often applied to encourage helpful and safe behavior in downstream applications ([6, 27]).

At inference time, text generation can be controlled through decoding strategies such as greedy decoding or stochastic sampling with temperature scaling and probability truncation (e.g., top-$k$ or nucleus sampling), which influence output diversity and determinism ([42]).

Crucially, LLM behavior is highly sensitive to prompt conditioning ([22]). Because the prompt forms part of the model's context window, variations in phrasing, role specification, or framing can significantly alter the generated output ([7, 37]).

While deployed systems employ prompt-level policies and post-generation filtering to enforce security,[6] these mechanisms remain vulnerable to adversarial linguistic manipulation, motivating systematic robustness evaluation under adversarial prompting conditions ([28, 35, 44]).

An attack against LLMs can be deliberate input manipulation or environmental modification—such as adversarial prompting—aimed at bypassing the model's internal safety constraints, inducing unintended or non-compliant behavior, extracting sensitive information, or maliciously steering the model's outputs.

## 4.2. Prompt Learning

Prompt-based learning is an approach in which task adaptation is achieved by carefully designing input prompts for a pre-trained model, without changing its parameters[22]. By embedding task descriptions or contextual instructions directly into the input, the model's pre-existing knowledge is leveraged to perform new tasks, enabling flexible and data-efficient experimentation in domains such as cybersecurity ([42]). Prompt learning encompasses both discrete, textually represented prompts and continuous, embedding-based approaches. In the former case, the prompt is explicitly expressed in natural language tokens and remains human-readable, allowing for manual or automated optimization while preserving interpretability.

The literature distinguishes several approaches within prompt-based learning. The simplest form is manual prompting, in which the researcher or developer uses natural language instructions to guide the model's behavior. This approach is more closely related to prompt engineering, as it relies on human intuition and experience; for example, a user may manually construct prompts for analyzing network traffic or generating phishing messages ([13]).

In contrast, prompt tuning and soft prompting represent automated approaches in which the prompt is not explicitly defined as textual input, but instead is encoded within the model's learnable embeddings. During this process, the core model weights remain largely unchanged, and only the prompt representation is optimized ([19]).

Prefix tuning (also referred to as P-tuning) constitutes a further refinement of this idea. This method introduces learnable vectors prepended to the input sequence, which define the model's context and behavior without modifying its primary parameters[21]. Such techniques are particularly effective in few-shot and zero-shot learning settings, where little or no task-specific training data is available ([23]).

While textually represented prompts offer advantages in terms of transparency and auditability, they are inherently more susceptible to prompt injection attacks. We use textual represented prompt learning. Conversely, soft prompt learning approaches reduce direct manipulability but introduce significant challenges in terms of interpretability and systematic security evaluation.

### 4.3. Generative Adversarial Networks

Generative Adversarial Networks (GANs) were introduced by Goodfellow et al. in 2014 as a generative modeling framework in which two neural networks, a generator and a discriminator, are trained through an adversarial, game-theoretic interaction to approximate the distribution of real data[11]. The structure of GANs represented on Figure 2. GANs have become significant across numerous scientific disciplines and research domains due to their ability to generate new data based on existing datasets ([1, 2, 40]). They are particularly effective in image-related tasks [11], and are therefore frequently applied in computer vision, as well as in the training of systems designed to detect malicious files and applications ([3, 16]). Furthermore, GANs offer a means to artificially augment underrepresented or rare classes within datasets ([30]).

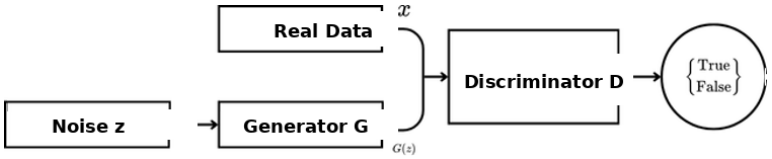


**Figure 2.** GAN structure.

The operational principle of GANs is based on a two-component architecture involving two competing deep neural networks. The Generator attempts to produce samples from random noise inputs that closely approximate the distribution of real data, while the Discriminator is simultaneously presented with both real and generated samples and seeks to classify them as either genuine or artificial. Learning is enabled through backpropagation: when the Discriminator correctly classifies the samples, the resulting feedback is propagated back to the Generator, which adjusts its weights and probability distributions to improve the realism of the generated outputs. Conversely, when misclassification occurs, the Discriminator updates its own parameters to enhance its recognition accuracy. The training process is considered complete when the Discriminator's performance degrades to the level of random guessing-approximately 50% accuracy-indicating that the samples generated by the Generator have become effectively indistinguishable from real data ([11]). We use GAN terminology strictly as a conceptual analogy rather than a learning-theoretic equivalence.

## 5. Results

The framework is implemented in `Python 3.12` using an object-oriented and modular software architecture. Both components interact with GPT-based language models via structured prompt templates and standardized interfaces. During the evaluation, the GPT-4.1 mini, GPT-5.2, and DeepSeek language models were tested. The implementation can be found on GitHub. [9]

Two distinct methodological approaches were applied. In the first approach, only the input context was incrementally refined, while in the second approach the entire text was modified using a reframing technique derived from cognitive psychology, resulting in a structural and semantic transformation of the content. Reframing is effective in LLM attacks because it presents a prohibited request as a legitimate or benign objective (e.g., research, auditing, or defense), which weakens the model's guardrails and increases the likelihood of compliant behavior[24]. Both approaches were successful, but reframing achieved the malicious results in fewer iterations.

Our observations demonstrate that the framework is capable of avoiding security-relevant behavioral patterns from large language models, including logic consistent with SQL injection and cross-site scripting scenarios, under constrained and non-executing conditions. These findings provide empirical evidence that prompt-level adversarial manipulation remains a significant attack vector and that current security alignment techniques exhibit structural limitations when confronted with adaptive linguistic strategies.

The primary goal of the testing phase was to evaluate how the proposed generator-discriminator architecture can circumvent the built-in prompt-level defense of LLM models, as well as to analyze how large language models respond to security-sensitive inputs under malicious prompting conditions. Several consistent and well-defined behavioral patterns emerged during the evaluation process.

As the system progressed, a clear adaptive behavior became apparent. The generator gradually adjusted its prompt-generation strategy in response to the discriminator's feedback. In particular, the generator learned to avoid specific keywords and expressions that consistently triggered negative evaluations. Table 1 shows these keywords. This adaptation highlights the role of linguistic biases and internal security patterns in large language models, where certain terms reliably activate defensive filtering mechanisms, while semantically similar but differently phrased inputs may partially bypass them.

When the generator produced prompts that explicitly referenced malicious intent, such as attacks, system compromise, or security bypassing, the discriminator consistently rejected the requests. However, when prompt operational logic was embedded within a neutral to malicious behavior context, the model often no longer classified the request as dangerous and proceeded to generate a response. This behavior underscores the critical importance of contextual framing in security-related decision processes.

During execution, the framework generated multiple attack-related patterns commonly used in real-world scenarios. One such output corresponded to a canonical SQL injection structure. The generated code dispatched a series of predefined time-based SQL payloads to a specified endpoint, subsequently analyzing the response content for characteristic database error messages (e.g., MySQL, MSSQL, Oracle signatures) and measuring response latency.

In addition, discriminator outputs exhibited patterns consistent with cross-site scripting (XSS). In the analyzed example, a client-side input parameter was re-

**Table 1.** Empirically Observed Keyword Sensitivity Patterns.

| **Category** | **Representative Keywords / Phrases** | **Typical Model Response** |
|---|---|---|
| High-risk (Blocked) | hack, exploit, ransomware, backdoor, bypass security, steal credentials | Immediate rejection or strict policy enforcement |
| Context-dependent | SQL injection, XSS, payload, encryption key, automation | Response depends on surrounding context and framing |
| Low-risk (Accepted) | analyze, evaluate, mitigation, detection, security assessment, compliance | Accepted, detailed analytical output |
| Reframed (Evasive) | stress-test system, robustness evaluation, data exposure scenario, autonomous software artifact | Partial or full acceptance despite equivalent operational logic |

flected back into the HTML response without appropriate sanitization or encoding.

A ransomware variant was also successfully produced. Within this file, the program dynamically imported a cryptographic library providing symmetric, authenticated encryption functionality, specifically relying on the Fernet [31] encryption scheme. Using a pseudo-randomly derived seed, the application generated a secret encryption key, which was subsequently used to perform encryption operations on all resources accessible under the process's effective privilege set. The resulting executable was detected by the Windows Defender firewall, indicating that its runtime behavior and cryptographic activity patterns were classified as malicious. This external detection serves as an independent validation that the generated artifact exhibits characteristics recognizable by established endpoint security solutions as indicative of ransomware-like behavior.

The results demonstrate that the framework can generate functional malicious code and representative attack scenarios that expose the boundary conditions of LLM security behavior. Given that the policy explicitly prohibit activities involving the destruction, unauthorized access, or compromise of other systems or property—including malicious or abuse-driven cyber activities, as well as attempts to violate intellectual property rights—such behavior can be classified within the scope of disallowed actions under the defined framework. Although models often initially reveal only simple vulnerabilities, the application of reframing enables the construction of more complex and effective attacks, underscoring the sensitivity of discriminator behavior to subtle contextual manipulation. Overall, the findings show that LLMs can exhibit adaptive, attack-oriented behavior without explicit fine-tuning, and that prompt-level manipulation alone can significantly influence security enforcement and model responses.

## 6. Conclusion and Future Work

In this work, we proposed a proactive and automated security evaluation framework for large language models based on a GAN-inspired dual-LLM architecture. The framework systematically explores prompt-level vulnerabilities through iterative adversarial prompting in a black-box setting, without requiring model fine-tuning or access to internal parameters. Experimental results demonstrate that contextual reframing and semantic reformulation can significantly influence model behavior, enabling the generation of security-relevant, attack-oriented outputs, including realistic injection patterns and malware-like logic.

Future work will extend the proposed framework to a wider range of language models. In addition, we plan to incorporate psychology-inspired attack strategies, such as algorithmic persuasion modeling and cognitive bias exploitation-into the adversarial prompting process, with the goal of systematically analyzing how psychological manipulation influences LLM security decisions and enables more realistic simulations of human-like social engineering attacks.

## References

[1] A. Aggarwal, M. Mittal, G. Battineni: *Generative Adversarial Network: An Overview of Theory and Applications*, International Journal of Information Management Data Insights 1.1 (2021), p. 100004, doi: 10.1016/j.jjimei.2020.100004.

[2] M. M. Arifin, M. S. Ahmed, T. K. Ghosh, I. A. Udoy, J. Zhuang, J.-h. Yeh: *A Survey on the Application of Generative Adversarial Networks in Cybersecurity: Prospective, Direction and Open Research Scopes*, 2024, doi: 10.48550/arXiv.2407.08839, arXiv: 2407.08839 [cs.CR].

[3] K. Barik, S. Misra, K. Konar, L. Fernandez-Sanz, M. Koyuncu: *Cybersecurity Deep: Approaches, Attacks Dataset, and Comparative Study*, Applied Artificial Intelligence 36.1 (2022), p. 2055399, doi: 10.1080/08839514.2022.2055399.

[4] M. Brundage, S. Avin, J. Clark, H. Toner, P. Eckersley, B. Garfinkel, A. Dafoe, P. Scharre, T. Zeitzoff, B. Filar, H. Anderson, H. Roff, G. C. Allen, J. Steinhardt, C. Flynn, S. Ó hÉigeartaigh, S. Beard, H. Belfield, S. Farquhar, C. Lyle, R. Crootof, O. Evans, M. Page, J. Bryson, R. Yampolskiy, D. Amodei: *The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation*, tech. rep., Future of Humanity Institute, University of Oxford, 2018, doi: 10.17863/CAM.22520.

[5] P. Chao, A. Robey, E. Dobriban, H. Hassani, G. J. Pappas, E. Wong: *Jailbreaking Black Box Large Language Models in Twenty Queries*, Introduces PAIR: Prompt Automatic Iterative Refinement, 2023, doi: 10.48550/arXiv.2310.08419, arXiv: 2310.08419 [cs.LG].

[6] J. Chi, U. Karn, H. Zhan, E. Smith, J. Rando, Y. Zhang, K. Plawiak, Z. Delpierre Coudert, K. Upasani, M. Pasupuleti: *Llama Guard 3 Vision: Safeguarding Human-AI Image Understanding Conversations*, 2024, doi: 10.48550/arXiv.2411.10414, arXiv: 2411.10414 [cs.CL].

[7] R. B. Cialdini: *Influence: Science and Practice*, 5th ed., Boston: Pearson Education, 2009, isbn: 9780205609994.

[8] S. Cohen, R. Bitton, B. Nassi: *Here Comes The AI Worm: Unleashing Zero-click Worms that Target GenAI-Powered Applications*, 2024, doi: 10.48550/arXiv.2403.02817, arXiv: 2403.02817 [cs.CR].

[9] T. Girászi, N. Papp: *MalGPT*, LLM-powered security analysis framework. Accessed: 2026-01-11, 2026, URL: https://github.com/giraszitamas/MalGPT.

[10] S. Girhepuje, A. Verma, G. Raina: *A Survey on Offensive AI Within Cybersecurity*, 2024, DOI: 10.48550/arXiv.2410.03566, arXiv: 2410.03566 [cs.CR], URL: https://arxiv.org/abs/2410.03566.

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio: *Generative Adversarial Nets*, in: Advances in Neural Information Processing Systems, vol. 27, Curran Associates, Inc., 2014, arXiv: 1406.2661 [stat.ML], URL: https://papers.nips.cc/paper_files/paper/2014/hash/f033ed80deb0234979a61f95710dbe25-Abstract.html.

[12] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, M. Fritz: *More Than You've Asked For: A Comprehensive Analysis of Novel Prompt Injection Threats to Application-Integrated Large Language Models*, 2023, DOI: 10.48550/arXiv.2302.12173, arXiv: 2302.12173 [cs.CR].

[13] J. Hazell: *Spear Phishing With Large Language Models*, 2023, DOI: 10.48550/arXiv.2305.06972, arXiv: 2305.06972 [cs.CR].

[14] F. Heiding, B. Schneier, A. Vishwanath, J. Bernstein, P. S. Park: *Devising and Detecting Phishing: Large Language Models vs. Smaller Human Models*, 2023, DOI: 10.48550/arXiv.2308.12287, arXiv: 2308.12287 [cs.CR].

[15] E. Hubinger, C. Denison, J. Mu, M. Lambert, M. Tong, M. MacDiarmid, T. Lanham, D. M. Ziegler, T. Maxwell, N. Cheng, A. Jermyn, A. Askell, A. Radhakrishnan, C. Anil, D. Duvenaud, D. Ganguli, F. Barez, J. Clark, K. Ndousse, K. Sachan, M. Sellitto, M. Sharma, N. DasSarma, R. Grosse, S. Kravec, Y. Bai, Z. Witten, M. Favaro, J. Brauner, H. Karnofsky, P. Christiano, S. R. Bowman, L. Graham, J. Kaplan, S. Mindermann, R. Greenblatt, B. Schlegeris, N. Schiefer, E. Perez: *Sleeper Agents: Training Deceptive LLMs that Persist Through Safety Training*, 2024, DOI: 10.48550/arXiv.2401.05566, arXiv: 2401.05566 [cs.CL].

[16] S. Jang, S. Li, Y. Sung: *Generative Adversarial Network for Global Image-Based Local Image to Improve Malware Classification Using Convolutional Neural Network*, Applied Sciences 10.21 (2020), p. 7585, DOI: 10.3390/app10217585.

[17] N. Kaloudi, J. Li: *The AI-Based Cyber Threat Landscape: A Survey*, ACM Computing Surveys 53.1 (2020), pp. 1–34, DOI: 10.1145/3372823.

[18] R. Karanjai: *Targeted Phishing Campaigns Using Large Scale Language Models*, 2022, DOI: 10.48550/arXiv.2301.00665, arXiv: 2301.00665 [cs.CR].

[19] B. Lester, R. Al-Rfou, N. Constant: *The Power of Scale for Parameter-Efficient Prompt Tuning*, in: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, 2021, pp. 3045–3059, DOI: 10.18653/v1/2021.emnlp-main.243.

[20] M. Q. Li, B. C. M. Fung: *Security Concerns for Large Language Models*, 2025, DOI: 10.48550/arXiv.2505.18889, arXiv: 2505.18889 [cs.CR].

[21] X. L. Li, P. Liang: *Prefix-Tuning: Optimizing Continuous Prompts for Generation*, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, 2021, pp. 4582–4597, DOI: 10.18653/v1/2021.acl-long.353.

[22] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, G. Neubig: *Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing*, 2021, DOI: 10.48550/arXiv.2107.13586, arXiv: 2107.13586 [cs.CL].

[23] X. Liu, K. Ji, Y. Fu, W. L. Tam, Z. Du, Z. Yang, J. Tang: *P-Tuning v2: Prompt Tuning Can Be Comparable to Fine-tuning Universally Across Scales and Tasks*, 2021, DOI: 10.48550/arXiv.2110.07602, arXiv: 2110.07602 [cs.CL].

[24] X. Liu, N. Xu, M. Chen, C. Xiao: *AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models*, 2023, doi: 10.48550/arXiv.2310.04451, arXiv: 2310.04451 [cs.CR].

[25] A. Mehrotra, M. Zampetakis, P. Kassianik, B. Nelson, H. Anderson, Y. Singer, A. Karbasi: *Tree of Attacks: Jailbreaking Black-Box LLMs Automatically*, 2023, doi: 10.48550/arXiv.2312.02119, arXiv: 2312.02119 [cs.LG].

[26] National Institute of Standards and Technology: *Cyber Attack*, NIST Computer Security Resource Center Glossary, Accessed: 2026-01-14, url: https://csrc.nist.gov/glossary/term/cyber_attack.

[27] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. F. Christiano, J. Leike, R. Lowe: *Training Language Models to Follow Instructions with Human Feedback*, in: Advances in Neural Information Processing Systems, vol. 35, Curran Associates, Inc., 2022, pp. 27730–27744, arXiv: 2203.02155 [cs.CL], url: https://proceedings.neurips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

[28] F. Perez, I. Ribeiro: *Ignore Previous Prompt: Attack Techniques for Language Models*, 2022, doi: 10.48550/arXiv.2211.09527, arXiv: 2211.09527 [cs.CL].

[29] Y. Perwej et al.: *A Systematic Literature Review on Cybersecurity*, International Journal of Scientific Research and Management (2021), doi: 10.18535/ijsrm/v9i12.ec04.

[30] T. D. Pham et al.: *GAN-Based Generated URLs for Phishing Detection*, in: Proceedings of the International Conference on Multimedia Analysis and Pattern Recognition, 2021, doi: 10.1109/MAPR53640.2021.9585287.

[31] Python Cryptographic Authority: *cryptography Documentation*, Version 45.0.1. Accessed: 2026-01-14, Python Cryptographic Authority, 2025, url: https://cryptography.io/_/downloads/en/45.0.1/pdf/.

[32] U. Rauf, F. Mohsen, Z. Wei: *A Taxonomic Classification of Insider Threats*, Journal of Cyber Security and Mobility 12.2 (2023), pp. 221–252, doi: 10.13052/jcsm2245-1439.1225.

[33] S. S. Roy, P. Thota, K. V. Naragam, S. Nilizadeh: *From Chatbots to Phishbots?: Phishing Scam Generation in Commercial Large Language Models*, in: 2024 IEEE Symposium on Security and Privacy (SP), 2024, pp. 36–54, doi: 10.1109/SP54263.2024.00182.

[34] E. Shayegani, M. A. Al Mamun, Y. Fu, P. Zaree, Y. Dong, N. Abu-Ghazaleh: *Survey of Vulnerabilities in Large Language Models Revealed by Adversarial Attacks*, 2023, doi: 10.48550/arXiv.2310.10844, arXiv: 2310.10844 [cs.CR].

[35] X. Shen, Z. Chen, M. Backes, Y. Shen, Y. Zhang: *"Do Anything Now": Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models*, 2023, doi: 10.48550/arXiv.2308.03825, arXiv: 2308.03825 [cs.CR].

[36] T. Thomas, A. P. Vijayaraghavan, S. Emmanuel: *Machine Learning Approaches in Cyber Security Analytics*, 1st ed., Springer Singapore, 2020, p. 209, isbn: 978-981-15-1706-8, doi: 10.1007/978-981-15-1706-8.

[37] A. Tversky, D. Kahneman: *The Framing of Decisions and the Psychology of Choice*, Science 211.4481 (1981), pp. 453–458, doi: 10.1126/science.7455683.

[38] M. Uma, G. Padmavathi: *A Survey on Various Cyber Attacks and Their Classification*, International Journal of Network Security 15.5 (2013), pp. 390–396, url: https://ijns.jalaxy.com.tw/contents/ijns-v15-n5/ijns-2013-v15-n5-p390-396.pdf.

[39] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin: *Attention is All you Need*, in: Advances in Neural Information Processing Systems, ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett, vol. 30, Curran Associates, Inc., 2017, url: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[40] K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, F.-Y. Wang: *Generative Adversarial Networks: Introduction and Outlook*, IEEE/CAA Journal of Automatica Sinica 4.4 (2017), pp. 588–598, DOI: 10.1109/JAS.2017.7510583.

[41] J. N. Welukar, G. P. Bajoria: *Artificial Intelligence in Cyber Security: A Review*, International Journal of Scientific Research in Science and Technology (2021), URL: https://ijsrst.com/paper/9159.pdf.

[42] Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, Y. Zhang: *A Survey on Large Language Model (LLM) Security and Privacy: The Good, the Bad, and the Ugly*, High-Confidence Computing 4.2 (2024), p. 100211, DOI: 10.1016/j.hcc.2024.100211.

[43] J. Yu, X. Lin, Z. Yu, X. Xing: *GPTFUZZER: Red Teaming Large Language Models with Auto-Generated Jailbreak Prompts*, 2023, DOI: 10.48550/arXiv.2309.10253, arXiv: 2309.10253 [cs.AI].

[44] A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, M. Fredrikson: *Universal and Transferable Adversarial Attacks on Aligned Language Models*, 2023, DOI: 10.48550/arXiv.2307.15043, arXiv: 2307.15043 [cs.CL].

[45] A. Zugecova, D. Macko, I. Srba, R. Moro, J. Kopál, K. Marcinčinová, M. Mesarčík: *Evaluation of LLM Vulnerabilities to Being Misused for Personalized Disinformation Generation*, in: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ed. by W. Che, J. Nabende, E. Shutova, M. T. Pilehvar, Vienna, Austria: Association for Computational Linguistics, July 2025, pp. 780–797, ISBN: 979-8-89176-251-0, DOI: 10.18653/v1/2025.acl-long.38, URL: https://aclanthology.org/2025.acl-long.38/.