An Empirical Study on Prompt Injection Attacks and Defences

Haritha Gunarathna
Department of Computer Engineering
University of Peradeniya
Kandy, Sri Lanka
e18118@eng.pdn.ac.lk

Denuwan Weerarathne
Department of Computer Engineering
University of Peradeniya
Kandy, Sri Lanka
e18382@eng.pdn.ac.lk

Nimuthu Wijerathne
Department of Computer Engineering
University of Peradeniya
Kandy, Sri Lanka
e18398@eng.pdn.ac.lk

Asitha Bandaranayake
Department of Computer Engineering
University of Peradeniya
Kandy, Sri Lanka
asithab@eng.pdn.ac.lk

Roshan Ragel
Department of Computer Engineering
University of Peradeniya
Kandy, Sri Lanka
roshanr@eng.pdn.ac.lk

Damayanthi Herath
Department of Computer Engineering
University of Peradeniya
Kandy, Sri Lanka
damayanthiherath@eng.pdn.ac.lk

***Abstract*—The recent surge of Large Language Models (LLMs) has revolutionized Natural Language Processing (NLP), showcasing remarkable abilities in text generation, translation, and comprehension. However, many vulnerabilities have been detected in LLMs. Prompt injection attacks specifically pose a significant threat to the security and integrity of LLMs. This paper presents a comprehensive literature review on this subject. We analyze existing research on prompt injection attack models, exploring their nature, attack models and prevailing taxonomies. We also delve into the impact of these attacks and discuss the current/ proposed defense strategies and mitigation. Finally, we discuss potential future directions in the area of prompt injection attacks. By this paper, we hope to contribute to the ongoing effort of securing LLMs and advancing responsible AI development.**

***Index Terms*—Large Language Models (LLMs), Prompt Injection Attacks, Adversarial Attacks, Security in LLMs, Multi-Tiered Defense, Jail-breaking, Indirect Prompt Injection**

I. Introduction

With its remarkable ability to produce text that is both coherent and contextually relevant, large language models (LLMs) have brought in a new era of natural language processing.This advancement hasn't, however, happened without scrutiny, and one major concern is that LLMs are vulnerable to adversarial attacks. Among these, prompt injection attacks have emerged as a particularly complex and potent form of manipulation.

Prompt injection attacks involve the strategic crafting of input prompts to deceive LLMs into producing unintended, unethical or biased outputs. Prompt injection attacks take use of the adaptability of language models that depend on user-provided prompts to generate responses, in contrast to classic adversarial attacks, which concentrate on altering specific tokens or words. This raises critical questions about the robustness and security of LLMs, especially as they are diverse in their applications, ranging from chat bots to content generation.

This review delves into the intricacies of prompt injection attacks within the broader context of adversarial attacks on LLMs. By examining the attack methods, impact, classifications and potential mitigation prevailing in the existing literature, this exploration aims to shed light on the evolving landscape of LLM security focusing on prompt injection attacks and contribute to the ongoing dialogue surrounding responsible AI development and deployment.

II. Methodology

In this section, we outline the methodology employed for the collection, testing, and documentation of LLM security and various prompt injection attacks and defenses. Given the primary objective of providing a comprehensive overview of the existing literature, our approach involved a systematic review of diverse sources, including arXiv preprints and platforms such as jailbreakchat.com, learnprompting.org, and owasp.com. Given the dynamic nature of the field and the frequent updates to chatbots and LLM APIs, we placed substantial emphasis on non-academic sources.
To identify prompt injections, we conducted keyword searches on Google, Google Scholar, arXiv, and Twitter (now X), employing terms such as ”prompt injection,” ”jailbreak,” ”large language models (LLMs),” ”direct prompt injection attacks,” ”indirect prompt injection attacks,” and ”LLM defense.” Complementing academic sources, we thoroughly reviewed content available on jailbreakchat.com, Twitter, and other websites. Addressing concerns related to the lack of peer review in non-academic sources as this is a relatively new subject, we adopted a two-step verification process. Initially, we sought multiple sources making the same claim, supported by credible screenshots illustrating the prompt injection mechanism. Subsequently, we conducted prompt injection tests in Bard, Bing Chat, GPT-3, and GPT-4 models. Due to the swift patching of vulnerabilities, these methods do

not consistently work without additional prompt engineering or may no longer be effective at the time of writing. As such, these vulnerabilities are pertinent primarily to future designs of chatbot and LLM interfaces.

III. Literature Review

A. LLMs, advancement, and usage

With their exceptional ability to comprehend and produce human language, large language models (LLMs) have taken their place as a key instrument in the Natural Language Processing (NLP) domain. Mathematical concepts based on deep learning architectures, like transformer models or recurrent neural networks (RNNs), are at the core of LLMs. LLMs can be categorized into various types, including transformer-based models (e.g. GPT-3), recurrent neural networks (e.g. LSTM), or even models using novel architectures such as the Transformer-XL, as shown in [18]. They have been extensively deployed across numerous fields and will be essential components in future communication networks [18]. Models (LLMs) like GPT-4 [6], LLaMA [11], and Bard [1], have dramatically transformed a wide array of applications with their exceptional ability to generate human-like texts [12]. Their applications span a broad spectrum of tasks such as machine translation, sentiment analysis, question answering, and text summarizing, opening new avenues for innovation and research in the field [18]. Large Language Models (LLMs) have been revolutionizing our lives in many ways, not just for practitioners and scholars but also for the general public and will continue to do so. This is evident by ChatGPT, which shortly after its release, gained immense popularity, attracting over 100 million users in a short period of time [7]. Furthermore, there is a constant stream of new models, including the more advanced GPT-4 [6], being introduced at a quick pace, and smaller white-box models [13].

Due to their superb generative capability, LLMs are widely deployed as the backend for various real-world applications called LLM-Integrated Applications. For example, Microsoft utilizes GPT-4 as the service backend for new Bing Search [1]; OpenAI developed various applications–such as ChatWithPDF and AskTheCode–that utilize GPT-4 for different tasks such as text processing, code interpreter, and product recommendation [16]. Other third parties can utilize LLMs for their diverse applications as well, including chatbots, assistants, and various other applications. With the establishment of the GPT Store in ChatGPT, custom models that align with the person's or organization's unique requirements and data have become publicly accessible, creating a marketplace of various AI tools designed for various applications [27].

B. Vulnerability Analysis of Generative AI models

The quick rise of LLMs and their expanded usage have given rise to many security concerns and vulnerabilities. In adversarial attacks [15], [25], which is a known threat to machine learning algorithms, carefully manipulated inputs can drive a machine learning structure to produce reliably erroneous outputs to an attacker's advantage [19]. Attacks can be targeted, seeking to change the output of the model to a specific class or text string, or untargeted, seeking only to result in an erroneous classification or generation as shown in [19]. Jailbreak, backdoor attacks, and complex data poisoning is shown as some examples of the broad spectrum of adversarial attacks in LLMs as per [12].

In [19] adversarial attacks are classified according to 5 concepts, Learning Structures, Injection source, Attacker access, Attacker type and the attack goal.

Based on the learning structure, attacks can be classified into uni-modal, multi-modal and additional attacks. Jailbreak attacks and prompt injection attacks are the two prevalent types of adversarial attacks on aligned uni-modal Large Language Models (LLMs), where uni-modal refers to focusing on manipulating a single type of data or output modality [19]. Multi-modal attacks, which are models that accept as input not only text, but additional modalities such as audio or images, are divided into 4, Manual attacks, systematic adversarial attacks, white-box attacks and black-box attacks. Other prevalent attacks can be classified under Additional Attacks as per the learning structure, which can be further divided into adversarial attacks in complex systems and earlier NLP attacks.

C. Prompt Injection Attacks and its Impact

Prompt injection attacks refer to a type of security threat where malicious users manipulate the prompts provided to Large Language Models (LLMs) or other AI systems to influence the generated outputs in unintended ways [1], [34].This method involves crafting input prompts in a manner that bypasses the model's safeguards or triggers undesirable outputs. It is the OWASP No.1 threat for LLMs [8]. There have been many adversarial effects caused due to prompt injection in generative AI models, some of which are listed below.

Generating text classified under prohibited scenarios 8 distinct prohibited scenarios from OpenAI's disallowed usage policy [6] that represents potential risks and concerns associated with the use of ChatGPT, was attempted to jailbreak in [21]. Comparatively [29] created and deployed a question set comprising 46,800 samples across 13 forbidden scenarios. Some forbidden scenarios taken into experimentation were Harmful content, Illegal activities, Political Lobbying, and Adult content. Both experiments showed that current LLMs and safeguards cannot adequately defend jailbreak prompts in those scenarios. Among the 13 forbidden scenarios tested in [29], Political Lobbying was found to be the most vulnerable to jail-breaking, followed by Pornography and Legal Opinion. With the prompts used in [21] it was found that Illegal activities, Fraudulent or Deceptive activity and Adult content were the easiest scenarios to be broken by jailbreak prompts. GPT-4 demonstrated a greater resistance against jailbreak prompts aimed at extracting prohibited content, compared to GPT-3.5-TURBO [21]. Additionally, it was concluded that jailbreak prompts easily achieve high success rates even in scenarios where initial resistance is observed, proving the capability of jail-breaking [29]. This was shown in [21] as well, as jailbreak prompts achieved a success rate of 74.6%, which is much higher than the rate of 29.0% for non jailbreak prompts.

Goal hijacking Goal Hijacking, also known as “Prompt Divergence” [35] attempts to redirect the LLM’s original objective towards a new goal desired by the attacker [19]. In other words, malicious users input prompts that hijack the original goal of LLM-integrated applications [17]. By deploying a novel framework ‘PROMPTINJECT’ in [14] to assemble prompts in the aim of goal hijacking, it achieved a success rate of 58.6%, showing that a malicious user can easily perform goal hijacking via human-crafted prompt injection. [19] Inspired by data poisoning and backdoor attacks, [36] introduce a novel concept of “Virtual” prompt injection attacks, focusing on goal hijacking, causing the model to answer a different question resulting in an answer of use to the attacker. Remarkably it was shown that, by contaminating only a small fraction of the instruction-tuning dataset, the attacker can influence the model’s behavior during inference when the model is queried about a specific target topic.

Prompt leaking/ System prompt extraction [14] defines prompt leaking as the act of misaligning the original goal of a prompt to a new goal of printing part of or the whole original prompt instead. A malicious user can try to perform prompt leaking with the goal of copying the prompt for a specific application, which can be the most important part of GPT-3-based applications. If an attacker can get access to the system prompt of a service provided by a company, they can build a clone of the service using the recovered system prompt making this prompt a valuable part of each system’s intellectual property [19]. Demonstrating this, [28] employs a leaked prompt to construct a mock LLM integrated application, observing a high degree of functional similarity between the two applications, implying that the leaked prompt can effectively replicate the capabilities of the original application as said before. A success rate of 23.6% was achieved through ‘PROMPTINJECT’ for prompt leaking [14], showing that it is notably more challenging than goal hijacking. However, [27] observed that deploying prompt injection attacks on 200+ custom GPTs yielded an alarming 97.2% success rate for system prompt extraction. They have defined system prompt injection as the act of deceiving custom GPTs into disclosing the designed system prompt, which is very much similar to prompt leaking.

File leakage File leakage is the act of stealing the designer-uploaded files used by the custom GPT for its specific purpose [27]. It has been identified that this vulnerability enables a malicious user to detect files uploaded by the custom GPT developer, including identifying the names and sizes of these files. The attacks on 200 custom GPTs, aiming file leakage, has a success rate of 100% highlighting an immediate need to address the risk of prompt injection.

Hallucination LLMs might make up facts (“hallucinate”), generate polarized content, or reproduce biases, hate speech, or stereo types [21], partially stemming from pre-training on massive crawled data sets. Even though one of the motivations for leveraging Reinforcement Learning from Human Feedback (RLHF) is to better align LLMs with human values and avert these unwanted behaviors [37], OpenAI reports that GPT-4 shows a tendency, to still hallucinate or generate harmful content [38].

Social Engineering/ Information gathering Attackers can manipulate the ”instructions” given to generative AI models, tricking unsuspecting individuals to reveal personal information, clicking malicious links, or sharing misinformation further, causing potential financial loss, reputation damage, or societal disruption [9]. As shown in [13] Indirect prompt injection could be leveraged to exfiltrate users’ data (e.g., credentials, personal information) or leak users’ chat sessions. This can be done in interactive chat sessions by persuading users to disclose their data or indirectly via side channels. ChatGPT’s ability to understand context, impressive fluency, and mimic human-like text generation could be leveraged by malicious actors [29]. The power of this approach lies in ChatGPT’s ability to generate text that aligns with the victim’s expectations, thereby increasing the likelihood of the victim complying with the request. [29]. Exfiltrating user’s private information is also listed as a threat by indirect prompt injection attacks in [17].

D. Characterization and Taxonomy of Prompt Injection Attacks

Current prompt injection attacks predominantly fall into two categories [34], direct and indirect prompt injection. In the case of direct prompt injection, an attacker has the capability to control their own engagement with the LLM. On the other hand, with indirect (or cross-domain) prompt injection, the attacker possesses the ability to manipulate the interaction of another user. [32]

Direct prompt injection attacks Direct prompt injection attacks [2], [34] operate on the premise of a malicious user directly injecting harmful prompts into the application inputs. The primary goal of these attacks is to manipulate the application into responding to a different query than its original intent. To achieve this, the adversary creates prompts, which can be natural language or communicate with LLM using cipher codes [24] like Caesar Cipher or Binary that can influence or negate the predefined prompts in the combined version, leading to the desired responses.

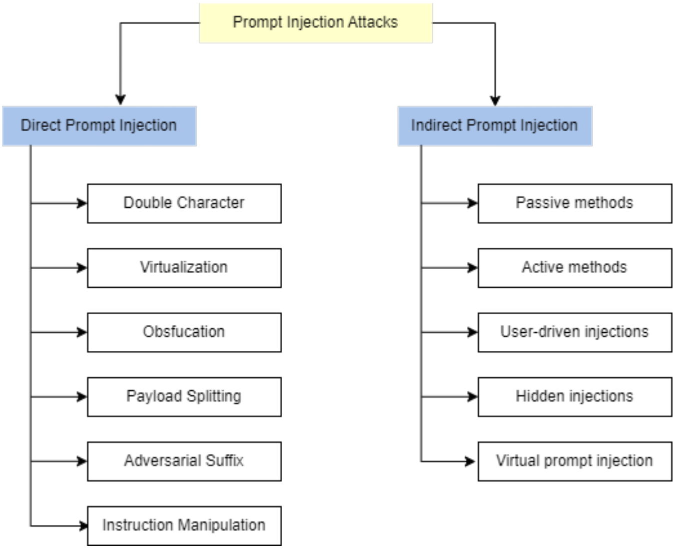

Fig. 1. Taxonomy of existing prompt injection attacks

We found that there exists a debate on whether jailbreak attacks [21] fall under direct prompt injection attacks as mentioned in [34]. In some papers [32], [34], jailbreak attacks are not considered as a prompt injection attack. Difference they highlight is that jailbreak attacks typically involve eliciting model-generated content that divulges training data specifics, potentially leading to privacy breaches. While direct prompt injection attacks focus on manipulating the LLM's output by appending malicious prompts to the input sequence, jailbreak attacks [3] target the model's training data specifics to extract sensitive information.But when referring to web pages like owasp.org [35] it clearly mentioned that jailbreak attacks are a type of direct prompt injection attacks.

According to [35] there are six sub classes of direct prompt injection attacks. They are Double Character, Virtualization, Obfuscation, Payload Splitting, Adversarial Suffix and Instruction Manipulation.

Jail-breaking Further research has been done on Jail-breaking as a separate topic. 10 patterns of Jail-breaking have been identified in [21], spread across 3 categories, Pretending, Attention Shifting and privilege escalation. It is evident from this study that pretending is the most prevalent strategy used by attackers to bypass restrictions while Attention shifting and privilege escalation are less frequently employed. Jailbreak prompts have been classified to 8 communities in [29], each demonstrating diverse and creative attack attempts in designing jailbreak prompts. It's found that Jailbreak prompts have evolved to be more stealthy and effective to conceal their malicious intent, evident by their reduced length, increased toxicity and semantic shift [29]. GPT4 demonstrates a greater resistance against jailbreak prompts aimed at extracting prohibited content, compared to GPT3.5 turbo, mainly because GPT4 has an improved ability to comprehend the output meaning [21]. Even though LLMs trained with RLHF (Reinforcement learning based on human feedback) show initial resistance to forbidden questions, they have weak resistance towards jailbreak prompts [29] External safeguards, such as OpenAI Moderation Endpoint and NeMo-Guardrails, demonstrate limited capability of identifying jailbreak prompts hence calling for stronger and more adaptive defense mechanisms [29].

Indirect prompt injection attacks The concept of Indirect Prompt Injection to compromise LLM-integrated applications was introduced in [13]. The first taxonomy and systematic analysis of the threat landscape associated with Indirect prompt injections in LLM-integrated applications was developed in [13] as well. In indirect prompt injection attacks, attackers inject malicious instructions into third party content, which when retrieved by an LLM-integrated application and ingested by the LLM, cause the LLM's output to deviate from the user's expectations [17]. As described in [13], a challenge associated with Large Language Models (LLMs) arises from the incorporation of retrieval, introducing ambiguity between data and instructions. Consequently, adversaries gain the capability to remotely influence other users' systems by strategically embedding prompts within data expected to be retrieved during inference.The LLM exhibits a reduced capacity to discern concealed instructions within the data, as the injection of prompts into the data becomes a viable operation. Hence it leads to Indirect prompt Injection attacks [13]. As LLMs progress, they demonstrate the capability to reinforce their response to a user prompt by surfing popular websites like wikipedia etc. [13] explains that it is done in a self-supervised manner employing in-context learning or using ReAct, utilizing Chain-of-Thought prompting. Hence with the progress of Large Language Models (LLMs) and their increasing capabilities, there is a growing susceptibility to indirect prompt injection attacks. Specially, LLM-integrated applications are vulnerable to indirect prompt injection attacks where malicious LLM misbehavior response instructions embedded within external content compromise LLM's output, causing their responses to deviate from user expectations [17]. There have been 4 methods introduced in [13] and publicly accepted, for indirect prompt injection attacks, being passive, active, user-driven injections and hidden injections. Passive methods rely on placement of malicious prompts or content inside a public source like websites, social media posts that might be read by an LLM. Alternatively, the malicious prompts could be actively delivered to the LLM, for example by sending emails containing prompts so that an email client enhanced with an LLM extension executes the prompt [13]. Sometimes the user himself can be tricked into injecting the malicious prompt themselves, resulting in an attack by an user-driven injection method. In hidden injections, attackers could use multiple exploit stages, where an initial smaller injection instructs the model to fetch a larger payload from another source [13]. Another indirect prompt injection attack method, as described in [41] is Virtual injection attacks, where the attacker manipulates the instruction tuning data of an LLM, so that in specific scenarios the model behavior is misaligned and provides outputs as if it was given additional instructions through a prompt. Indirect prompt injections have resulted in many adverse threats like Information gathering, fraud and spreading malware [13].

E. Novel Attack models

Many prompt injection attack models have been designed and introduced in research papers, some being entirely novel attacks that draw inspiration from the general context [14], [28], while others are frameworks that try to systematize and formalize prompt injection attacks [16].

A novel prompt injection framework called 'PROMPTINJECT' was proposed in [14], for mask-based iterative adversarial prompt composition and it was shown how GPT3, the most widely deployed language model in production, can be easily misaligned by simple handcrafted inputs. In particular, two types of attacks are demonstrated, goal hijacking and prompt leaking. The framework consists of 2 parts, the base prompt and the attack prompt.

A novel black box prompt injection attack technique called HOUYI was introduced in [28], drawing inspiration from traditional web injection attacks. HOUYI is compartmentalized into three crucial elements: a seamlessly-incorporated pre-constructed prompt, an injection prompt inducing context partition, and a malicious payload designed to fulfill the attack objectives. Leveraging HOUYI, previously unknown and severe attack outcomes were unveiled, such as unrestricted arbitrary LLM usage and uncomplicated application prompt theft. HOUYI was deployed on 36 actual LLM-integrated applications, where it was successful on 31 applications. HOUYI consists of a prompt refinement with dynamic feedback, ultimately, helping it to output a collection of successful attack prompts.

As a different approach (and the first of its kind) a framework was proposed in [16], to systematize and formalize prompt injection attacks. Under the framework, different prompt injection attacks essentially use different strategies to craft the compromised data prompt based on the clean data prompt, injected instruction of the injected task, and the injected data of the injected task. Existing attacks are considered to be special cases in this framework. Based on this framework, a new prompt injection attack was designed by combining existing attack strategies, and the results show that this combined attack outperforms others. A defense framework was also introduced along with. The introduced framework-inspired attack was proved to be consistently effective for different target and injected tasks. Another interesting finding of this study [16] was that in general, the attack is more effective when the LLM is larger (or more powerful), and the reasoning that a larger LLM is more powerful in following the instructions and thus is more vulnerable to prompt injection attacks, is suspected.

F. Defense Mechanisms

In the conducted extensive review over the available literature on the domain of Prompt Injection Attacks, it was found that there have been many attempts to come up with a relatively accurate and unbreakable defense mechanism against these types of attacks. Following is a brief description of the findings.

In paper [17], authors propose different types of Defense mechanisms based on two different aspects. One is White-box defense. This mechanism leverages the access to work with model parameters and architecture along with the training process. Other mechanism is called Black-box Defense. In this, there is no access to the model inner workings, parameters and training process, but it is based on prompt learning techniques such as usage of border strings, data-marking, usage of multi-turn dialogue and in-context learning (few-shot learning). White-box Defenses mainly includes fine tuning the model through adversarial training. Main motivation of these defense mechanisms is the LLMs' inability to distinguish between external content and instructions. [17]

As another defense mechanism, another paper [4], [17] propose a harmful filter be used in front of the inference model. The harm filter can be another LLM or another instance of the same LLM. The harm filter LLM is primed to identify the toxicity of the incoming prompt. The user input is formatted into a specific format before feeding into the harm filter. This method has shown significant evaluation metrics when tested with prominent LLMs like GPT3 and Llama2.

It is also possible to use more traditional ways of defending like input validation and output validation. Regex patterns, whitelists, blacklists can be used to identify a potential prompt injection attack [4], [5].

Post-prompting can also significantly improve the defense layer strength of LLMs. What this does is passing the system/user instruction after passing the external (potentially malicious) content first into the LLMs. This way it can work against attacks which convince the model to ignore all previous instructions [14].

Some other notable and traditional defense mechanisms are using XML tags (use XML tags to cover the external content), Random sequence enclosure (enclosing the external content with Random sequences) and sandwich defense (enclosure of the external content between the two instances of the same original prompt). All these methods try to enhance the LLMs ability to identify the line between provided instructions and the external content [1].

A different approach was taken by the author Xuchen [40] in his paper where he signs the user instruction prompt in a certain way that even the same prompt, when exposed to the model as two prompts (signed and unsigned), the model is able to identify as two different entities. This way, unless the signing signature is released to the public, a properly trained LLM is capable to differentiate between malicious instructions and user instructions and act accordingly.

Other than this there were many other recently introduced defense mechanisms as well in addition to the traditinal defense mechanisms we found when reviewing many other papers. [22], [26], [30], [31], [33], [39].

IV. Conclusion

With the recent rise of Large Language Models or LLMs, they have made their way into the society, directly and as the back end of production grade applications and user interfaces. With this different adversarial attacks have emerged against LLMs along with various defence strategies to prevent and mitigate those.

In this study we have reviewed numerous peer-reviewed and preprint papers to map the existing literature into a clear and comprehensive introduction of Prompt Injection Attacks and Defenses. We start the review with an introduction to our empirical study, and then briefly describe the methodology we followed to search for existing literature on the topic. An important factor to consider is that most of the academic research papers we reviewed were not peer reviewed (preprints), mainly due to the fact that the topic of LLMs and Prompt Injection Attacks is relatively new, at the time of writing. Because of this and due to the fact that this is a rapidly advancing topic, we had to heavily rely on non-academic sources like private blogs of notable domain experts, twitter threads, and documentations.

We then move onto explaining briefly about LLMs, usage and their recent rapid advancements at the time of writing this article. Then we give an overview of the security threats present in the Generative AI domain. We next address the focused security threat of this paper, Prompt Injection Attacks. We start off with and brief introduction and then in length describes the impact of these attacks to the security, privacy and ethics of generative AI and its end users. Next, from all the information gathered from reviewing number of papers, we provide a taxonomy of prompt injection attacks as well as a brief introduction into other controversial attack types. We also provide some novel prompt injection attack strategies, that we came across. Thereafter we move onto describing some traditional as well as some solid, recently evolved defense mechanisms which has shown significant resilience against many prompt injection attacks, and has proved to be working with prominent LLMs like LLama2 and GPT4.

References

[1] "Learn Prompting: Your Guide to Communicating with AI," learnprompting.org. https://learnprompting.org/docs/prompt_hacking/injection (accessed Feb. 06, 2024).

[2] "LLM Vulnerability Series: Direct Prompt Injections and Jailbreaks — Lakera – Protecting AI teams that disrupt the world.," www.lakera.ai. https://www.lakera.ai/blog/direct-prompt-injections (accessed Dec. 25, 2024).

[3] "Jailbreak Chat," www.jailbreakchat.com. https://www.jailbreakchat.com/ (accessed Feb. 09, 2024).

[4] "The Dual LLM pattern for building AI assistants that can resist prompt injection," simonwillison.net. https://simonwillison.net/2023/Apr/25/dual-llm-pattern/#dual-llms-privileged-and-quarantined (accessed Feb. 02, 2024).

[5] "Exploring Prompt Injection Attacks," NCC Group Research Blog, Dec. 05, 2022. https://research.nccgroup.com/2022/12/05/exploring-prompt-injection-attacks/ (accessed Feb. 09, 2024).

[6] OpenAI, "Usage policies," openai.com, Mar. 23, 2023. https://openai.com/policies/usage-policies (accessed Jan. 27, 2024).

[7] K. Hu, "ChatGPT Sets Record for Fastest-Growing User Base," Reuters, Feb. 02, 2023. Available: https://www.reuters.com/technology/chatgpt-sets-record-fastest-growing-user-base-analyst-note-2023-02-01/ (accessed Feb. 09, 2024).

[8] OWASP, "OWASP foundation, the open source foundation for application security," owasp.org, 2023. https://owasp.org/ (accessed Jan. 12, 2024).

[9] "Gemini - chat to supercharge your ideas," gemini.google.com. https://gemini.google.com/app (accessed Feb. 09, 2024).

[10] "ChatPDF - Chat with any PDF!," www.chatpdf.com. https://www.chatpdf.com/ (accessed Feb. 09, 2024) .

[11] "Llama," Llama. https://llama.meta.com (accessed Feb. 09, 2024).

[12] Y. Liu et al., "Prompt Injection attack against LLM-integrated Applications," arXiv (Cornell University), Jun. 2023, doi: 10.48550/arxiv.2306.05499.

[13] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, "Not what you've signed up for: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection," arXiv (Cornell University), Feb. 2023, doi: 10.48550/arxiv.2302.12173.

[14] F. Perez and I. Ribeiro, "Ignore previous prompt: Attack techniques for language models," arXiv (Cornell University), Nov. 2022, doi: 10.48550/arxiv.2211.09527.

[15] L. Xu, Y. Chen, G. Cui, H. Gao, and Z. Liu, "Exploring the universal vulnerability of prompt-based learning paradigm," Findings of the Association for Computational Linguistics: NAACL 2022, Jan. 2022, doi: 10.18653/v1/2022.findings-naacl.137.

[16] Y. Liu, Y. Jia, R. Geng, J. Jia, and N. Z. Gong, "Prompt injection attacks and defenses in LLM-Integrated applications," arXiv (Cornell University), Oct. 2023, doi: 10.48550/arxiv.2310.12815.

[17] J. Yi et al., "Benchmarking and defending against indirect prompt injection attacks on large language models," arXiv (Cornell University), Dec. 2023, doi: 10.48550/arxiv.2312.14197.

[18] H. Yang, K. Xiang, H. Li, and R. Lu, "A Comprehensive Overview of Backdoor Attacks in Large Language Models within Communication Networks," arXiv (Cornell University), Aug. 2023, doi: 10.48550/arxiv.2308.14367.

[19] E. Shayegani, M. A. A. Mamun, F. Yu, P. Zaree, D. Yue, and N. Abu-Ghazaleh, "Survey of vulnerabilities in large language models revealed by adversarial attacks," arXiv (Cornell University), Oct. 2023, doi: 10.48550/arxiv.2310.10844.

[20] H. Li et al., "Privacy in large language models: attacks, defenses and future directions," arXiv (Cornell University), Oct. 2023, doi: 10.48550/arxiv.2310.10383.

[21] Y. Liu et al., "Jailbreaking ChatGPT via Prompt Engineering: An empirical study," arXiv (Cornell University), May 2023, doi: 10.48550/arxiv.2305.13860.

[22] D. Yip, A. Esmradi, and C. F. Chan, "A novel evaluation framework for assessing resilience against prompt injection attacks in large language models," arXiv (Cornell University), Jan. 2024, doi: 10.48550/arxiv.2401.00991.

[23] R. Pedro, D. Castro, P. Carreira, and N. R. D. Santos, "From Prompt Injections to SQL Injection Attacks: How Protected is Your LLM-Integrated Web Application?," arXiv (Cornell University), Aug. 2023, doi: 10.48550/arxiv.2308.01990.

[24] Y. Yuan et al., "GPT-4 Is Too Smart To Be Safe: Stealthy Chat with LLMs via Cipher," arXiv (Cornell University), Aug. 2023, doi: 10.48550/arxiv.2308.06463.

[25] A. Zou, Z. Wang, J. Z. Kolter, and M. Fredrikson, "Universal and transferable adversarial attacks on aligned language models," arXiv (Cornell University), Jul. 2023, doi: 10.48550/arxiv.2307.15043.

[26] J. Piet et al., "JaTMo: Prompt Injection Defense by Task-Specific Finetuning," arXiv (Cornell University), Dec. 2023, doi: 10.48550/arxiv.2312.17673.

[27] J. Yu, Y. Wu, S. Dong, M. Jin, and X. Xing, "Assessing prompt injection risks in 200+ custom GPTs," arXiv (Cornell University), Nov. 2023, doi: 10.48550/arxiv.2311.11538.

[28] X. Suo, “Signed-Prompt: A new approach to prevent prompt injection attacks against LLM-Integrated applications,” arXiv (Cornell University), Jan. 2024, doi: 10.48550/arxiv.2401.07612.

[29] X. Shen, Z. Chen, M. Backes, Y. Shen, and Y. Zhang, “‘Do Anything Now’: Characterizing and evaluating In-The-Wild Jailbreak prompts on large language models,” arXiv (Cornell University), Aug. 2023, doi: 10.48550/arxiv.2308.03825.

[30] Y. Yao, J. Duan, K. Xu, Y. Cai, E. Sun, and Y. Zhang, “A survey on Large Language Model (LLM) Security and Privacy: The Good, the bad, and the ugly,” arXiv (Cornell University), Dec. 2023, doi: 10.48550/arxiv.2312.02003.

[31] A. Robey, E. Wong, H. Hassani, and G. J. Pappas, “SmoothLLM: Defending large language Models against jailbreaking Attacks,” arXiv (Cornell University), Oct. 2023, doi: 10.48550/arxiv.2310.03684.

[32] A. H. Salem, A. Paverd, and B. Köpf, “Maatphor: Automated variant analysis for prompt injection attacks,” arXiv (Cornell University), Dec. 2023, doi: 10.48550/arxiv.2312.11513.

[33] P. Rai, S. Sood, V. K. Madisetti, and A. Bahga, “GUARDIAN: A Multi-Tiered Defense architecture for thwarting prompt injection attacks on LLMs,” Journal of Software Engineering and Applications, vol. 17, no. 01, pp. 43–68, Jan. 2024, doi: 10.4236/jsea.2024.171003.

[34] Y. Liu et al., “Prompt Injection attack against LLM-integrated Applications,” arXiv (Cornell University), Jun. 2023, doi: 10.48550/arxiv.2306.05499.

[35] Erfan Shayegani, Yue Dong, and Nael Abu-Ghazaleh. 2023. Plug and pray: Exploiting off-the-shelf compo nents of multi-modal models. arXiv preprint arXiv:2307.14539.

[36] J. Yan et al., “Virtual prompt injection for Instruction-Tuned large Language models,” arXiv (Cornell University), Jul. 2023, doi: 10.48550/arxiv.2307.16888.

[37] Y. Bai et al., “Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback,” arXiv (Cornell University), Apr. 2022, doi: 10.48550/arxiv.2204.05862.

[38] OpenAI, “GPT-4 Technical Report,” arXiv (Cornell University), Mar. 2023, doi: 10.48550/arxiv.2303.08774.

[39] A. Helbling, M. Phute, M. S. Hull, and D. H. Chau, “LLM self defense: By self examination, LLMs know they are being tricked,” arXiv (Cornell University), Aug. 2023, doi: 10.48550/arxiv.2308.07308.

[40] X. Suo, “Signed-Prompt: A new approach to prevent prompt injection attacks against LLM-Integrated applications,” arXiv (Cornell University), Jan. 2024, doi: 10.48550/arxiv.2401.07612.

[41] S. Rossi, A. Michel, R. R. Mukkamala, and J. B. Thatcher, “An early categorization of prompt injection attacks on large language models,” arXiv (Cornell University), Jan. 2024, doi: 10.48550/arxiv.2402.00898.