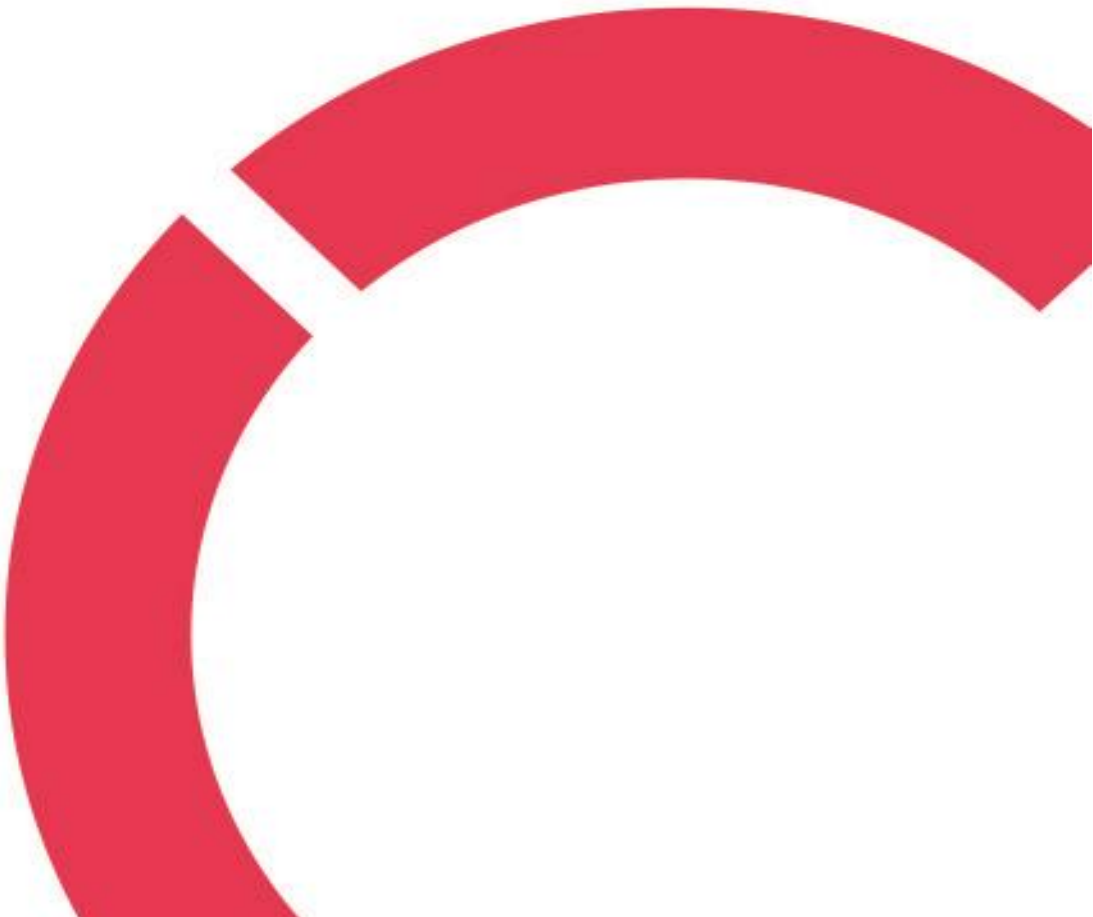
Hridoy Shaha

UNDERSTANDING PROMPT INJECTION ATTACKS IN WEB BASED LLM LLM APPLICATIONS AND BASIC MITIGATION STRATEGIES

Thesis
CENTRIA UNIVERSITY OF APPLIED SCIENCES
Bachelor of Engineering, Information Technology
May 2025



ABSTRACT

Centria University of Applied Sciences	Date	Author
	May 2025	Hridoy Shaha
Degree programme		
Bachelor of Engineering, Information Technology		
Name of thesis		
UNDERSTANDING PROMPT INJECTION ATTACKS IN WEB BASED LLM APPLICATIONS AND BASIC MITIGATION STRATEGIES		
Centria supervisor		**Pages**
Henry Paananen		**36 + 5**

Natural language processing systems are being revolutionized by large language models (LLMs), which allow them to do complex tasks requiring advanced technology, generate text that mimics human speech, and more. However, their connection to online apps has also made serious security flaws like prompt injection attacks possible. The thesis here explores the underlying mechanics of these prompt injection attacks, which it divides into two types: direct prompt injection attacks and indirect prompt injection attacks. In direct prompt attacks, criminals attach a command directly to users' inputs, and in indirect methods, they obtain outside references to embed the malicious payloads. The study shows how prompt injection can compromise a system's integrity, expose confidential data, and destroy user faith through specific case studies of hacked AI assistants. In response, this work presents an architecture for multiple mitigations through robust authentication and authorization mechanisms, high-level validation, and sanitization of input, and continuous surveillance to detect and act upon any malicious activity. In this study, it promises future indications in adaptive security measures and makes a direct analysis of the strengths and weaknesses of existing defenses to inform adaptive security measures that will change pace with new threats. In addition, this study propels development in the direction of creating applications that are faster and therefore more reliable and secure based on LLMs, thus ensuring gain from such innovative systems without compromising online security.

Key words

Artificial Intelligence, Deep Learning, Generative Artificial Intelligence, Large Language Model, Machine Learning, Natural Language Processing.

CONCEPT DEFINITIONS

Artificial Intelligence (AI)

A set of technologies called artificial intelligence (AI) lets computers carry out many sophisticated tasks including data analysis, recommendations, and more; the ability to see, understand and translate spoken and written language.

Deep Learning (DL)

A part of machine learning called deep learning mimics the sophisticated decision-making ability of the human brain using multilayered neural networks known as deep neural networks. Most of the artificial intelligence (AI) applications in our daily lives are powered by some kind of deep learning.

Generative Artificial Intelligence (GenAI)

Sometimes known as gen AI, generative artificial intelligence (AI) is that which can produce original material including text, pictures, movies, audio, or software code in reaction to a user's request or prompt.

Large Language Model (LLM)

A kind of foundation model called large language models (LLMs) is trained on enormous datasets, enabling them to comprehend and produce natural language as well as other kinds of content to carry out a wide range of activities.

Machine Learning (ML)

Machine learning (ML) is a subfield of artificial intelligence (AI) that aims to let machines and computers mimic the way people learn, to carry out tasks independently, and enhance their performance and accuracy by means of experience and exposure to more data.

Natural Language Processing

Natural language processing (NLP) is a field of study of computer science and artificial intelligence (AI) that uses machine learning to let computers understand and interact with human language.

ABSTRACT

CONCEPT DEFINITIONS

CONTENTS

LIST OF FIGURES

1 INTRODUCTION

In recent years, artificial intelligence technologies and large language models (LLMs) have greatly affected their areas by allowing ever-evolving improvements in the process of automation and decision-making. Large language models based on transformer architectures like OpenAI's GPT and Google's Gemini have dramatically changed the natural language processing (NLP) application systems, and they have achieved remarkable capabilities in many areas, including customer service, education, health, creative content creation, etc. Nevertheless, having moved increasingly into the web-based environments, a plethora of new security or ethical issues arise, which could destabilize the trust and thus the safety of the system. From these challenges, prompt injection attacks have emerged as a serious matter, able to take advantage of the very system that allows LLMs to be so powerful and easy to use for the user. Even though their contextual knowledge has phenomenal performance potential, it is also easily vulnerable to injection attacks. The consequence of such attacks can be wide-ranging, compromising data trustworthiness, user privacy, and secure application of industry-wide deployment. The purpose of this thesis is to investigate the underlying mechanisms of prompt injection attacks, namely, what makes them susceptible and how they conduct an attack on the reasoning of the LLM-based web applications (i.e., the logic they are based on). In addition, it aims to give an overall sense of general mitigation strategies to counteract such threats. This thesis is based on an investigation of real-life case studies as well as discussions from theoretical and practical perspectives to make a contribution towards the line of literature on how to guarantee protection of LLMs for web applications. The contribution of this work is to the growing use of LLM-based systems for the task in question. With increasing uptake of these technologies comes increased risk of misuse of their capabilities. Protection against prompt injection vulnerabilities, while important for the defense of systems, also becomes relevant in fostering trust that will ultimately build into the power of AI. Briefly, the work of the current thesis is about the leading class of prompt injection attacks on models of large language models (LLMs) for web-scale applications and introduces the first line of defense against such attacks. In this study, firstly, the type of prompt injection attack will be characterized and classified, and then the mechanism and some illustrative examples will be presented in an everyday scenario. The following paragraphs will address the domains of influence of input validation, authentication, impairments in capabilities, and monitoring techniques as one fundamental aspect of the exploitation of LLM-based systems. More broadly, this thesis contributes to a broader understanding of the security of LLMs, in turn providing the building blocks for more secure and robust AIs in a dynamic digital world.

2 OVERVIEW OF LARGE LANGUAGE MODELS

Large Language Models (LLMs) form the basis for all present artificial intelligence and are the heart of future paradigms for human language understanding and generation. These models, built on transformer architectures, leverage massive datasets and computational power to achieve unprecedented levels of natural language understanding and generation. LLM development is motivated by the ongoing advancement of deep learning, such as the optimization of self-attention mechanisms and the application of large-scale pre-training methods, which allows LLMs to achieve performance on a large variety of downstream tasks with few fine-tuned parameters (Zhao, Zhou & Li 2023.) Over the past decade, LLMs have changed from statistical-based and neural linguistic models to intelligent generative models that can perform in-context, instruction-based learning and high-level reasoning (Minaee et al. 2024). Their scalability and generalizability to other applications enable them to be applied across a variety of applications, spanning web, automated assistant, or as the architecture of machine learning-driven decision-making (Minaee, Mikolov, Nikzad 2024).

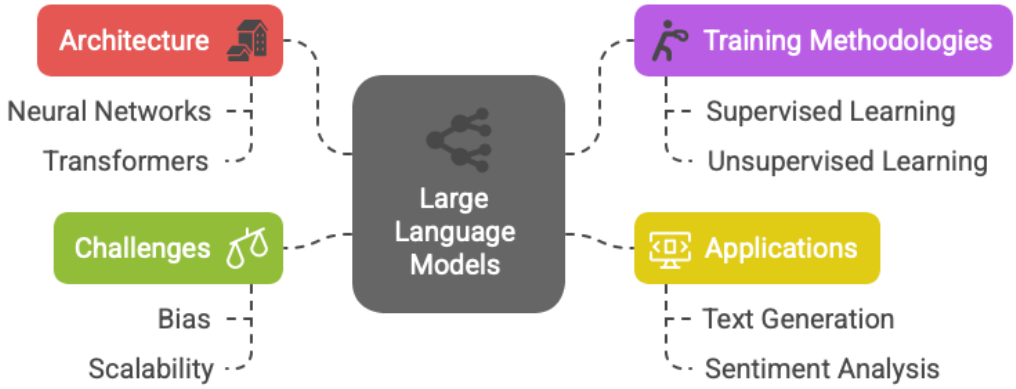


Figure 1. Comprehensive Overview of Large Language Models (NapkinAI 2025)

2.1 What are Large Language Models

Large language models are deep learning models of a transformer architecture that generate human-like text in a fluent way from vast amounts of training data. They are characterised by the scale of their size-billions or trillions of parameters; they are characterised by their high complexity of language

structure and abstract reasoning (Zhao et al. 2023.) For example, in these kinds of models, pretraining is done on several heterogeneous data sources (i.e., literature, journal articles, and webpages), and thus they have a satisfactory level of linguistic understanding. In contrast with classical models of language which were trained on a specific task, LLMs can be adapted for a wide range of tasks without heavy fine-tuning (Liu, Tang & Sun 2025.)

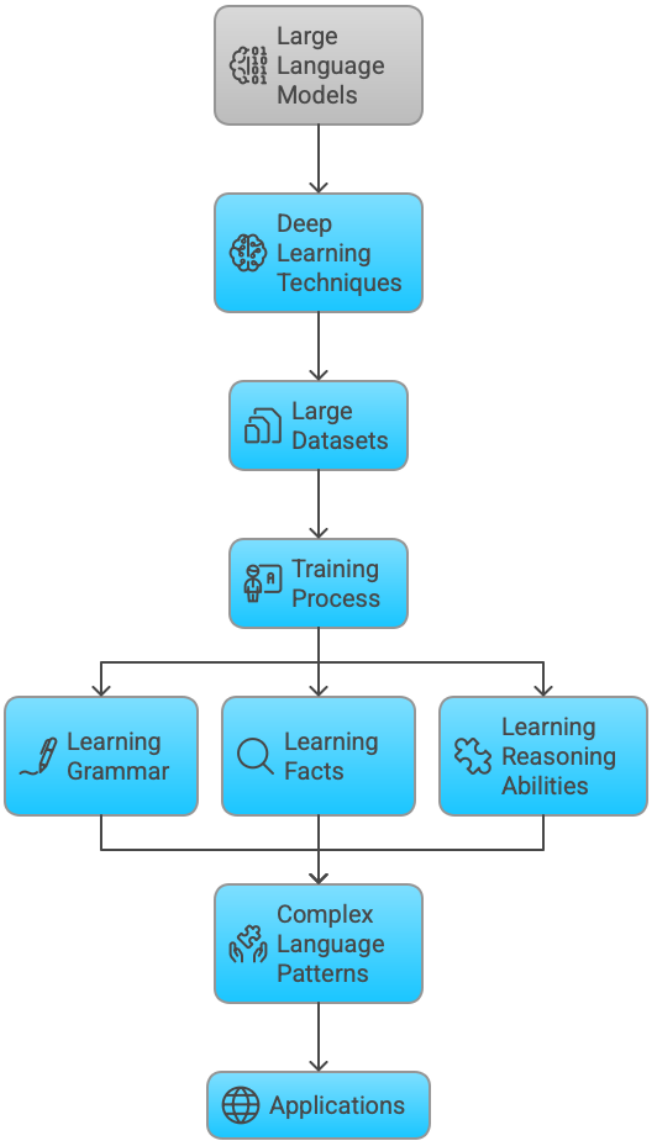


Figure 2. Large Language Models Process (NapkinAI 2025)

The emergent capability is one of the least perturbative of the LLMs, in direct relation to in-context learning and zero-shot reasoning. This ability is due to size scaling, in which, even in the absence of discrete small improvements, larger model sizes are empirically associated with actual performance improvements (Berti, Giorgi & Kasneci 2025.) For instance, models like GPT-4, DeepSeek, and Gemini show impressive capacity to decode instructions, to learn complex tasks, and to produce grammatically plausible, contextually relevant text (Bubeck, Chadrasekaran & Eldan 2023).These functionalities also grant LLMs a high degree of phenotypic plasticity, allowing LLMs to extend their capabilities beyond simple text generation tasks like machine translation, summarization, and code generation (Berti et al. 2025).

While, the recent novel technologies have made many advances, at the same time, problems at this time are that they are computationally inefficient, difficult to interpret, and have ethical issues. Model training has to be computationally expensive and power-hungry, thereby raising sustainability and portability issues (Bolón-Canedo, Morán-Fernández, & Cancela 2024.) Furthermore, this reliance on a vast number of samples can lead to bias that will persist in the generated outputs, and that bias must be scrutinized with respect to fairness and accuracy. Addressing these challenges is crucial for us to ethically design and use LLMs in real-life tasks (Sakib, Islam & Pathak 2024.)

2.2 Applications of LLMs in Web-Based Systems

LLMs have been used for diverse application scenarios in web-based applications, including enhancing user interaction, automating processes, and enhancing content production. The ability of LLMs to learn and respond to human-like queries has greatly boosted their applicability to customer service chatbots, virtual assistants, and agent-based search engines. Now a days, in customer service, human interaction can now be personalized and maintained at an exceedingly low level with all-around service provided by LLMs. Mainly, LLMs have an equally great role to play in the content recommendation systems that serve to tailor user experiences to user preferences and user history (Raza, Jahangir & Riaz 2025). Apart from the conversational AI, LLMs are used for content moderation and sentiment analysis. They make possible the detection of abusive content and spam elimination along with user engagement analysis on social media and e-commerce application platforms. Moreover, they can support simultaneous bidirectional language translation capabilities, decreasing barriers to communication across languages and achieving uninterrupted interlingual communication (Kumarage, Bhattacharjee & Garland 2024.) Large Language Models (LLMs) could

summarize text and generate content that is rich in content information that has already been used in research and news aggregation. This allows the users to quickly filter through thousands of pages of text for information (Zhang, Jin & Meng 2024).

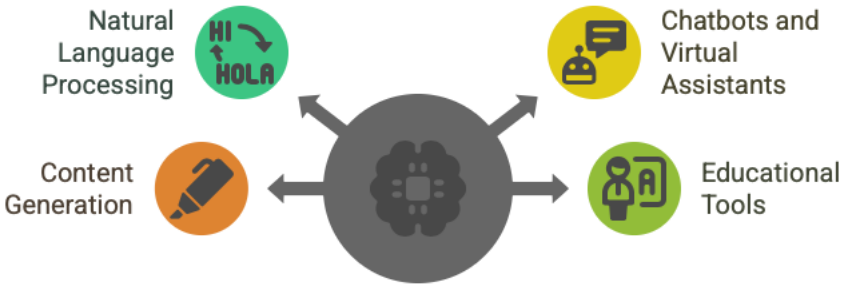


Figure 3. Applications of LLMs (NapkinAI 2025)

However, the web application embedding of LLMs is far from easy. Security flaws, such as prompt injection attacks, can be exploited to attack model outputs and attack/break the system. Moreover, to guarantee accountability when it comes to utilizing AI, there is an ethical responsibility that comes with deception and imbalanced information that needs to be put in place with very careful checks (Liu, Deng & Meng 2024) It is also an essential requirement in large steps towards bringing into practice at least a part of the potential of LLMs and the capacity to cope with the danger it poses (Madani, Tavasoli & Astaneh 2025).

2.3 Integration of LLMs into Web Applications

The implementation of the use of LLMs in web applications should also be addressed in system architecture, scalability and security. The Application Programming Interface driven architectures are also one of the most popular models to make use of LLMs so that the models are exposed on-line as a service to mine the user queries online in real-time (Tzachristas 2024.) This methodology can be easily combined with the conventional web platforms, which means that the app developers have ability to take advantage of the LLM features does not require replicating major infrastructure changes. However, effective API management and low latency are key challenges of deployment at scale (Hau, Hassan & Zhou 2025.)

The designing security measures to prevent adversarial manipulation is another key aspect of integration. Please note that the article describes several types of attacks on LLMs, such as prompt injection and data poisoning, that can generate inappropriate or harmful outputs (Abdali, Anarfi & Barberan 2024.) To prevent these kinds of risks, developers use methods like input sanitization, role-based access control, and monitoring in real-time to identify and eliminate possible threats. Also, because of fine-tuning, the models become more reliable by contracting their responses based on guidelines and ethics (He, Qiu & Zhang 2024.)

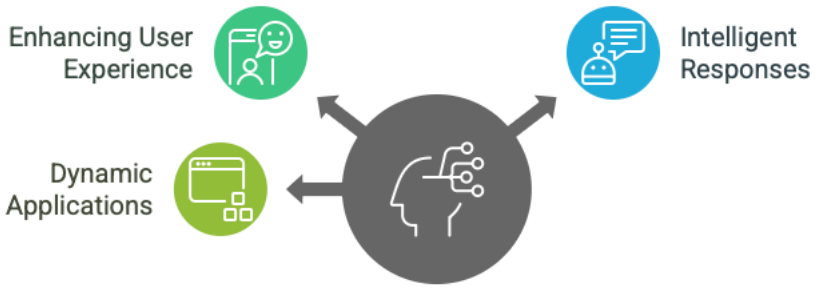


Figure 4. Integrating LLMs into Web Apps (NapkinAI 2025)

The scalability is another critical factor in web based LLM integration. As demand for systems increases, it is necessary to be able to support concurrent requests while maintaining response quality and comptaional efficiency. Methodologies (Model Distillation and Model Quantization), which reduce the demand of resources without performance degradation (Yao, Jin & Shah 2024.) Additionally, adaptive training in learning mechanisms allows LLMs to adapt their outputs in real time and, therefore, their accuracy and the user satisfaction evolves over time (Lee, Whang & Lee 2023).

In general, the good integrating of LLMs in web applications depends on the combination of performance optimization, security strengthening, and ethical adherence. Due to the ongoing research, these challenges must be tackled in order to finally prove the potential that the LLMs can offer to the next generation of web services (Madani et al. 2025.)

3 PROMPT INJECTION ATTACKS

Prompt injection attacks have been identified as a critical security vulnerability in systems that employ large language models (LLMs) as web application elements. Such attacks take advantage of the vulnerability of LLMs to produce malignant instructions in the deceptively benign manner of a natural utterance, and to produce the opposite of the natural output for the final LLMs (Liu, Deng & Li 2023.) As LLMs are deployed to a variety of applications such as customer service or content generation, the likelihood that an attacker may bypass it through indirect prompting instead of intended prompting has increased (Hackett, Birch & Trawicki 2025.) Most significantly, the power of these attacks stems from their ability to reflect those very factors that make LLMs so powerful, including in-context learning and response generation over time (Huang, Zhan & Wong 2025.) Consequently, it is of paramount concern to comprehend the mechanism of prompt injection attack, in order to prevent the reliability and trustworthiness of applications such as AI-based systems (Liu, Jia & Geng 2024).This chapter explains the reason, fundamental principles of operation, types and examples of prompt injection attacks along with context on its broader impact on subsequent counter measures.

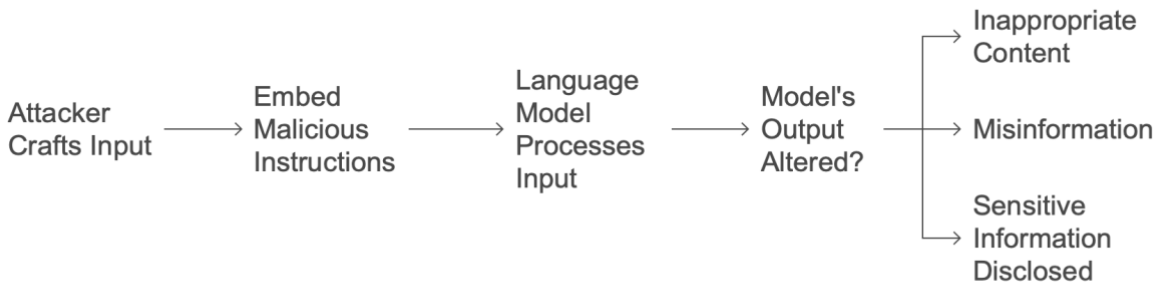


Figure 5. Prompt Injection Attack Process (NapkinAI 2025)

3.1 Definition and Concepts

Prompt injection attacks are attacks in which input prompts are intentionally crafted to drive LLMs away from their behavior. Essentially, an attacker creates some input, which includes commands that are malicious and are integrated into the normal data, so that the model performs unintended

behaviors. The present idea is based on the sensitivity of LLMs, which take for granted all textual input according to the learned patterns and can also be misled by deliberately disguised malicious instructions. In essence, the attack exploits that the model does not distinguish between a regular user input and a injected adversarial command, thereby masking the boundary between legitimate instructions and malicious instructions (Liu et al. 2023.) The proposal has been contrasted with conventional injection attacks like SQL injection, in which formatted data is exploited to change the computer program behavior (Pedro, Castro & Carreira 2023). Through this foundational concept, the researchers can develop better defences that reduce the effects of such injections.

3.2 Mechanics of Prompt Injection Attacks

Prompt injections are attacks on the large language models based on altering the structure of prompts strategically for coercing the model into generating unintended outputs (Pan, Wong & Yuan 2025). Attackers execute such attacks in a multi-phase way wherein the first step involves getting acquainted with the context of the specific application, followed by design to create a malicious payload after which it goes to refining iteratively the injected command based on model feedbacks. The underlying mechanics are similar to traditional injection attacks where the adversary injects harmful commands into a stream of input to that system to evade security filters (Liu et al. 2024.) The capitalizing on how LLMs rely on input context as well as sequence continuity, an attacker can alter the ultimate output without compulsory alterations to the insides of the system's core instructions. The above process identifies that LLMs can be rendered vulnerable to slight manipulations in their input pipeline (Qiang, Zhou & Zhu 2023.) These attacks also leverage the ability of the model to combine different sources of information, including what the user inputs and default prompts. Such a critical understanding will be important in establishing any defense against prompt injection (Pan et al. 2025.)

In the initial phase, context inference, the attacker attempts to analyze the site very closely application targets to try to capture how user inputs are and system prompts are combined. In this way, the adversary examines the application that connects the already prepared prompts with a dynamic query from a user revealing the weaknesses in their merging phase (Liu et al. 2024). This phase is critical because it is the starting point for an attack, revealing the limits and boundaries of the input handling by the system itself. Initially context inference as in the previous case; with the attacker's reconnaissance methods, one might be an automated query and capture logging among responses so that it can draw a map of the internal composition of the prompt (Zhan, Fang & Panchal 2025.) What

one is to understand in context also defines which parts of the input can be altered without triggering alarms at the moment. A very thorough mapping enables attackers to specify payloads that combine seamlessly within the legitimate input. By comprehensively capturing the operational context, the attacker will be able to predict how injected content might influence model behavior. This extensive analysis of the context is the precondition for the actual injection phase (Pan et al. 2025.)

The next stage from context inference is payload generation, wherein the attacker crafts a malicious textual string intended for embedding into the regular prompt.This payload is set up in such a way that it manipulates a particular one of the potentially exploitable faults identified through the context analysis, normally by including commands that override existing instructions. Other example commands could include directives such as "ignore previous instructions" or any other phrases that motivate the model to stray from its intended task (Liu, Zhou & Liu 2024.) The various ways used by the attacker to embolden the malicious content among the benign text include the use of obfuscation or escape characters. These techniques ensure that the payload injected is undetectably scrutinized to be harmful by most rudimentary security filters. But payload creation is also about going through a cyclic process wherein initial payloads are adjusted according to the first outputs of the model (Zhang, Sullivan & Jackson 2025.) Iterative refinement allows one to ensure that a payload is effective in changing the meaning of the prompt but is also stealthy for detection. Therefore, it is a challenge both creative and technical in nature and forms the basis upon which a successful attack rests (Ke, Lai & Fang 2025.)

This has presented above the last stage in the incredible procedure of prompt injection attacks, which is called the feedback phase, in which the attacker judges the model output and modifies the payload accordingly. Here in lies the part of the adversary that continuously monitors the output responses generated by the LLM for any change or indication that the malicious payload is taking effect (Pan et al. 2023). Once that does not match the required modification, it has to be further adjusted and tried again, thus creating a self-improving loop. Feedback from the system is crucial as it informs the attacker about the effectiveness of the injection and whether additional modifications are necessary (Aryal, Gupta & Abdelsalam 2024.).

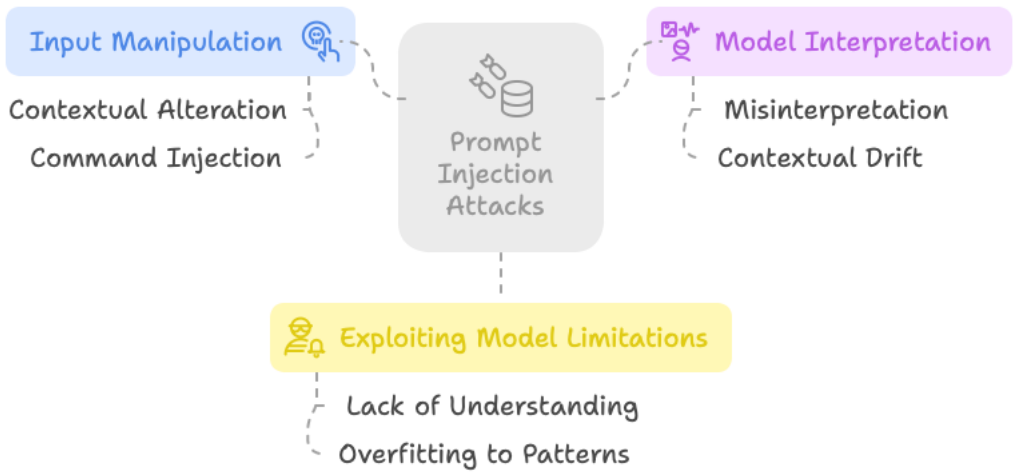


Figure 6. Mechanics of Prompt Injection Attacks on LLMs (NapkinAI 2025)

The feedback phase is not only improves the success rate of the attack but also makes sure that the model generates the lowest possible overt diversion in its output. In a nutshell, the iterative phase signifies the effectiveness of prompt injection attacks to adjust according to the adaptive defences of the system (Pan et al. 2025.)

The basic operations of a prompt injection attack can be categorised into three sequential processes: context inference, payload generation, and iterative feedback. The each phase has exploits coordinated vulnerabilities in LLM-integrated systems that allow attackers to incorporate malicious commands that deviate into unintended functionality of the application (Liu, Deng & Li 2023.) Similar to the prompt injection attacks in other areas, it is a new way to benefit from their natural language processing capabilities by those modern LLMs. The detailed contextual study, sophisticated payload design, and iterative enhancement allow attackers to reach a very high success rate in compromising outputs from LLMs (Liu et al. 2024.) Knowledge of such as imaginative mechanics is necessary to develop mitigation strategies to detect and thus successfully counter such attacks (Pan et al. 2025). This complete promise of prompt injection mechanics is the launchpad for the later discussion on case studies as well as further defense measures in this thesis.

3.3 Direct Prompt Injection

Direct prompt injection is that attack method where attacks occur when a malicious command directly appends to user's original input, so that it would change the general prompt given by feeding input into the large language model. This relies on the model's inherent trust on the input; the combination of harmful commands with legitimate queries ends up leading the LLM into interpreting and executing unintended commands (Liu et al. 2024.) Thus, in many cases, attackers only specify the commands like "ignore previous instructions" or any other commands that force the model to deviate from the activity normally expected by it. The simplicity of this attack lies exactly in that; the malicious payload is visibly attached to the user query without any intermediate processing, which makes it one straightforward if not powerful attack channel (Greshake, Abdelnabi &Mishra 2023).

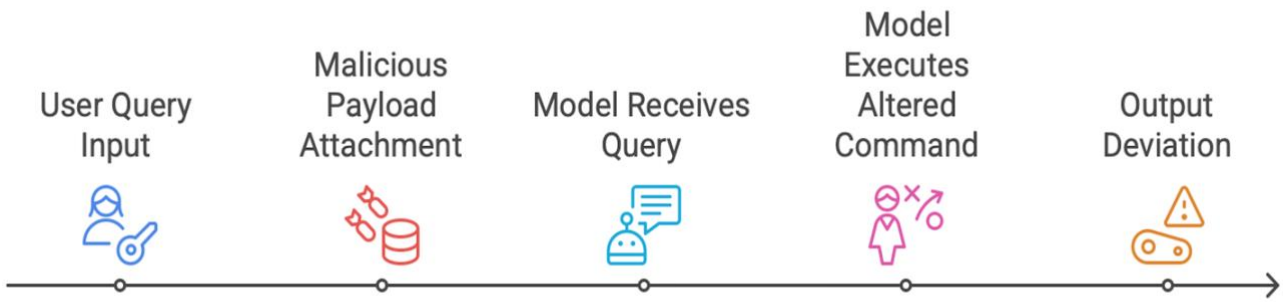


Figure 7. Direct Prompt Injection Attack (NapkinAI 2025)

The emphasis of the attacks turns now to their effects on system security and integrity. Through sucessful injection of commands to the model, the LLM will often respond to requests producing outputs perhaps a sensitive answer or mimic the attacker's intended output format It highlights the urgency of implementing quality input validation and filtration mechanisms against malicious text floodings into the system controls (Liu et al. 2024.) Furthermore, a direct manner of attack reveals a major opening in present defenses; even the most minimal of errors in prompt construction could be used as leverage to bring an entire system down (Greshake, Abdelnabi & Mishra 2023). Overall, direct prompt injection demonstrates how critical it is to deign LLM applications with layered security measures to mitigate simple attack vectors but which cause a lot of damage (Mathew 2024).

3.4 Indirect Prompt Injection

Indirect prompt injections are indirect methods of injecting harmful commands into an application as opposed to sending them directly with the user's input. The attacker will hide the harmful command in external data or background content, which the LLM later incorporates into its main prompt. The model may then undesirably incorporate the adversary's commands as they seem to be embedded in unrelated or additional information (Greshake et al. 2023.) It uses techniques like obfuscation by character substitutions or masking semantic encoding to such a malicious input so that it can be passed through the normal security filters.This approach exploits those situations when the model cannot differentiate between the valid target instructions and the actually injected external data, lowering the probability of detecting the injected command (Liu 2024.)

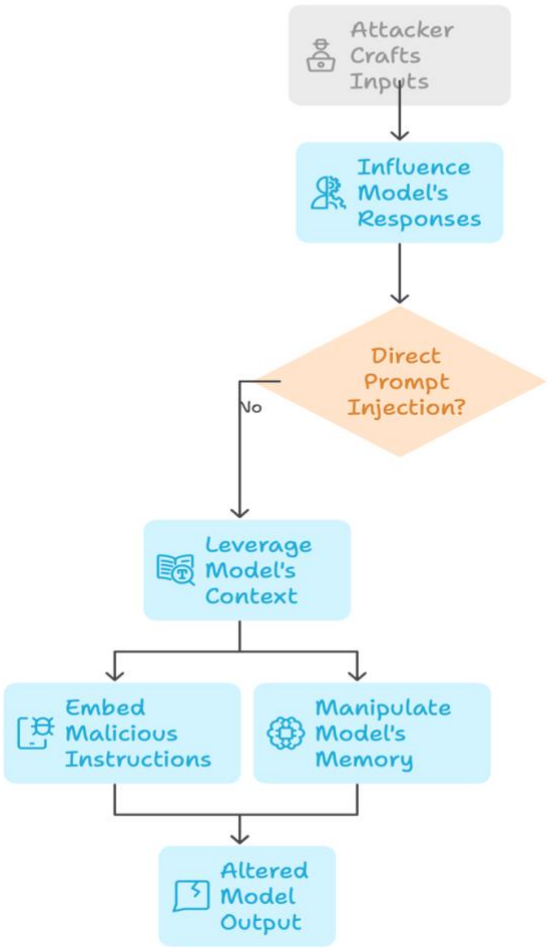


Figure 8. Indirect Prompt Injection Attacks (NapkinAI 2025)

Prompt injection attacks in large language models can be very dangerous because they can change the model's answers without being noticed. The traditional input validation techniques usually fail to capture it because the malicious payload is not directly visible in the user input (Khomsky, Maloyan & Nutfullin 2024.) This position usually boosts the chances of exploitation going undetected. This types of prompt injection attacks can severely affect the confidentiality and integrity of the large language models system, as the system's manipulable output may be exploited. In combating this type of prompt injection, techniques for detection need to be advanced in their consideration of contextual shifts and integrations within the data-escorting architecture to camouflage threats (Liu 2024.)

3.5 Examples of Attacks in Web-Based Large Language Model Applications

The prompt injection attacks are a reality in the present of which puts into play the very real risk associated with web based LLM applications (Liu et al. 2024). We see a case where the attacker added a command to what was a customer’s support chatbot request which in turn made the model to return out of band information instead of the customer’s query response. Also we have reports of the use of escape characters and other obfuscation methods that disrupt the normal prompt structure and get the LLM to present a command as part of a different set of instructions (Hao,Yang & Lin 2024.) These reports of attacks show that although it is possible to design applications with strong security features, there is still a weak point when user input is not properly validated or separated from the main instructional set (Liu et al. 2024).) Also brought out are the issues that for which continuous monitoring and adaptive defenses like real time violation of prompt structure standards are required (Pedro et al. 2025).

For example we see in the case of an educational platform which has interactive learning environments where LLMs are used we have reported issues of input prompt injection which in turn made the model to give out biased or inappropriate answers (Yamamura & Ganguli 2024). That point brings out the issue of threat prompt injection which we see to be a great danger to user safety and application robustness in particular within public platforms which have millions of users interaction with LLMs daily. Also we note how a simple question may cause great changes in the system’s response when the issue of improper protection is present (Greshake et al. 2023).

3.6 Impact on Web Security and User Trust

Prompt injection attacks are threaten the web security in several respects through the degradation of the robustness of the applications based on LLMs by maliciously injecting prompts into the systems which is probably evoke outputs containing the private information or invoke unintended actions (Liu et al. 2023). Therefore, in LLMs input data are manipulated by the attackers to evade the imposed safety checks and override the desired functionality of the application. The above situation severely compromises the integrity of the system as a whole (Rababah, Wu & Kwiatkowski 2024.) This is because of the types of vulnerabilities make users uncertain and degrade their confidence in LLMs in domains like finance and healthcare, where data security is usually a top priority (Abdali et al. 2024).

Furthermore, such prompt injection attacks contribute to the crea-tion of a much broader mindset where LLMs appear as, by nature, untrustworthy or even dangerous in critical applications which discourages the potential users from adopting these technologies (Rababah et al. 2024). This hesitance could block the uses and integration of LLM-based systems in several places, including commerce and government services, where proper functioning is paramount to operational success (Pantha, Ramasubramanian & Gurung 2024.) If such as attacks could be periodically emphasized, it would be require organizations to very diligently consider security issues in order to maintain public confidence and sustainable LLM-driven applications over time (Abdali et al. 2024).

Besides the fact of that the prompt injection attack itself highlights the need for developers to understand the different classes, inner working, and possible ramifications of such a attacks so that they can implement their countermeasures. If the vulnerabilities have been identified and classified, the developers can devise countermeasures that would not only protect against malicious acts but also protect against loss of users trust (Pedro et al. 2025). The countermeasures are include but are not limited to sophisticated input sanitization, tightened the authentication mechanisms, and dynamic system monitoring for abnormal patterns of the prompt, before the actual attack takes place (Mathew 2024). Finally, it is very important to handle such issues to protect both the actual operation of LLM-integrated apps and the trust of end users (Divakaran & Peddinti 2024.)

4 BASIC MITIGATION STRATEGIES

The developing integration of large language models into web-based applications requests consideration to security vulnerabilities like provoke infusion assaults to protect framework astuteness (Liu, Deng & Meng 2024). LLMs generate human like content however stay helpless to antagonistic assaults which can result in unauthorized get to systems and the spread of wrong data along with uncovering private information. These models require defensive measures which will anticipate their abuse and keep up both us ability and productivity (Abdali et al. 2024.) The fundamental moderation procedures comprise of the input approval and sanitization along with client verification and authorization whereas too implementing capability confinements for LLMs and executing checking and logging frameworks and designer security conventions. The chapter looks at these moderation procedures completely and assesses their effectiveness along with the execution challenges they show (Dong, Mu & Jin 2024.)

4.1 Input Validation and Sanitization

From the perspective of a primary line of defense in LLMs, one should validate and sanitize the user inputs so that it conforms to the secure treatment of receiving the inputs into the LLMs (Dong et al. 2024). It can excludes any possible input considered malicious or intrusive and thereby denies the possible adversarial manipulation. There are many kinds of validating an input methods. It involves such as pattern matching, defining regular expressions, and heuristic-based filtering to detect and neutralize harmful prompts before they reach to the model (Ishibashi & Shimodaira 2023.) Also, the input sanitization such as automatic tokenization and context-aware filtering, impedes attempts by potential intruders to engage in prompt injections that would be override internally defined instructions within the system (Pedro et al. 2023). However, these methods seem to lose their potency fast against the attackers intelligence in creating smart prompts that utilize the ambiguity of natural language and differences in encoding to bypass standard filtering (Esmradi et al. 2024).

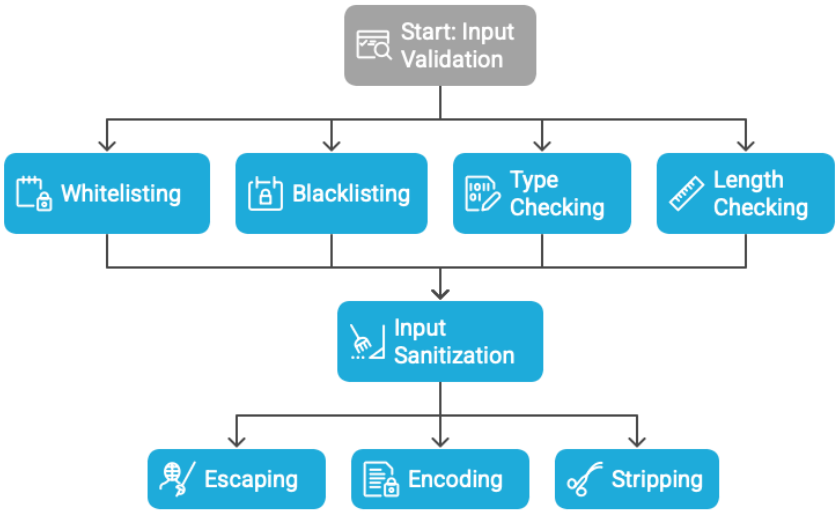


Figure 9. Input Validation and Sanitization Techniques (NapkinAI 2025)

4.1.1 Techniques for Input Validation

The whitelisting by filtering the input is the first and the foremost way to protect against injection attacks on web applications interacting with LLMs (Yao et al. 2024). The whitelisting can optionally be joined by some form of deterrent technique or simply do its work to bar the input pattern before injection into the model. The static whitelisting could be supplemented with matching or pattern search as offered by regular expressions that would block off dangerous known entry types before model processing (Ishibashi & Shimodaira 2023.) In general, it uses a sophisticated method for different types of machine anomaly detector which, to some extent, accepts or disregards such undetectable inputs by users depending on learned behavior patterns (Pingua et al. 2024). Another way to perform context-aware validation and semantic structural matching of prompts is to create prompt addition resources to detect subtle adversarial inputs. On the other hand, token-level clearing mechanisms include masking or swapping for an innocuous token sequence before sending to the LLM. Together these provide more comprehensive protection schemes that would become harder to circumvent or invalidate with the advanced forms of attacks (Zhang et al. 2025.)

4.1.2 Common Challenges

On one hand, the first attack prompt injection defenses face such obstacles as the appreciably evolving attack techniques and the apparent balancing act between security and usability (Yao et al. 2024). On the one hand, such rigid filtering rules might block some legitimate queries from reaching the model, thereby restricting its functionality. On the other hand, overly relaxed filtering might actually allow successful prompt injection attempts to pass by, adding to the security risk (Ishibashi & Shimodaira 2023.) Furthermore, indirect prompt injection attacks will generally prove to be very difficult even to notice, as the attacker tries to embed poisonous circuitry through contextual information that the LLM should listen to (Zhang et al. 2025). For that, a hybrid architecture could be recommended that combines rule-based filtering and machinelearning anomaly detection (Esmradi et al. 2024).

4.2 User Authentication and Authorization

The security of applications interfaced to the LLM, granting users access to interact with them, is therefore important as a preventive measure (Yao et al. 2024). User authentication mechanisms are primarily used to verify users identities before allowing them access to LLM-based systems while preventing unauthorized prompt injections (Ishibashi & Shimodaira 2023). Thus, permission policies would confer access rights to users and prevent non-trusted users from accessing any sensitive operations of the model (Pingua et al. 2024). Other measures that would protect LLMs against any damage initiated by attackers are implementing strong authentication and access control mechanisms within the application itself (Zhang et al. 2025).

Figure 10. User Authentication Methods (NapkinAI 2025)

4.2.1 Implementing Secure Authentication

To secure means of the authentication are multi-mode of authentication, biometric verification, and cryptographic key-based of access control (Esmradi et al. 2024). The multi-way authentication makes it possible to add a layer of security with another verification factor, for example, password plus one-time code so no one can take easily access control (Yao et al. 2024). The biometrics authentication on characteristics of either fingerprint or face restricts the access to the LLMs system only for the verified users (Ishibashi & Shimodaira 2023). The type of authentication through cryptographic keys involving digital certificates and public key infrastructure (PKI) gives a very secure method of verifying the users identity (Pingua et al. 2024).

4.2.2 Role Based Access Control

Role-Based Access Control is a fundamental security measure that restricts access to the system based on the roles and responsibilities of its users in a way that these functions are secured against unauthorized execution. Thus, a sensitive operation can only be executed by an authorized actor (Esmradi el al. 2024). It is here that organizations can create counterparts for roles such as administrators, developers, and end users, granting adequate permissions to all roles so they may fulfill the principle of least privilege (Zhao et al. 2023). In this way, the attack surface is reduced due to unauthorized users not being able to use important functions or access confidential data in LLM-integrated applications (Ishibashi & Shimodaira, 2023). While further granting RBAC of fine-grained control over system resources, monitoring, auditing, and modifying access policy for resources becomes easier as per evolving organitional demand (Pingua et al. 2024). Well defined roles limit the risk of attacks such as prompt injection because attackers are much less likely to gain access to privileged functionalities (Zhang et al. 2025). The introduction of RBAC further enhances the resilience of the system as anomalies concerning access would be swiftly detected and contained (Esmradi et al. 2024). Therefore, control of access using RBAC will empower the developers to maintain a vigorous security posture with respect to both the integrity of LLM output and the privacy of user data (Zhao et al. 2023).

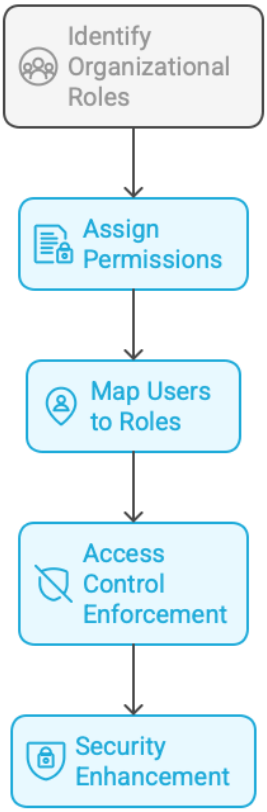


Figure 11. Role Based Access Control Process (NapkinAI 2025)

So far, in practice, the implementation of RBAC involves setting up a complete policy framework that defines precisely each user role, the permissions pertaining to it, and the security measures for enforcing the said permissions (Ishibashi & Shimodaira 2023). Such a framework should, on dynamism, allow regular reviews and updates so that information can be injected as they relate to changes in organi-zational structure and the evolution of the threats that it interacts with (Pingua et al. 2024). Integration with multifactor authentication only makes RBAC more robust in that even users with legitimate roles must confirm their identity using other security measures (Zhang et al. 2025). This com-bined with user activity monitoring by Continuous Monitoring Tools provide a perfect combo for an RBAC as it monitors user activity and flags any abnormality away from their ordinary behavior by giving alerts to the Security team (Esmradi et al. 2024). It keeps track of all access attempts along-side those actions that accountably fall in the realm of specific kind of function that could be applicable for forensic investigations into data breach situations, supporting data protection compliance (Zhao et al. 2023).

Moreover, the adoption of RBAC allows for an environment of accountability, wherein every user is tracked for his/her actions in the system that can always be traced back to his/her role (Ishibashi & Shimodaira 2023). For both internal reviews and external regulatory audits, the detailed audit of RBAC is invaluable (Pingua et al. 2024). Thus, role-based access control is among the most important configurations securing LLM-based applications and ensuring access is only to those with the right authority, with a view to exploiting this very least (Zhang et al. 2025).

4.3 Limiting Large Language Model Capabilities

The restrictions on the capabilities of a large language model will reduce the risk of some types of adverse events inflicted by such types of adversarial attacks as prompt injections a sort of manipulation that limits the types of actions that a model can take even when it is compromised (Esmradi et al. 2024). In this way, it will impose tighter operational limitations to keep the developed model's replies well within the safety limits, even under such attacks of very sophisticated kind (Zhao et al. 2023). This has to do with both technical adjustments to the architecture of the model and also well-thought control mechanisms that limit the outputs of the model (Ishibashi & Shimodaira 2023). Limits on capability not only reduce the attack surface but also maintain the general integrity and soundness of web based applications (Pingua et al. 2024). Such restrictions will minimize the chance of creating a potential opening for abuse while creating a safer operating environment for LLMs (Zhang et al. 2025).

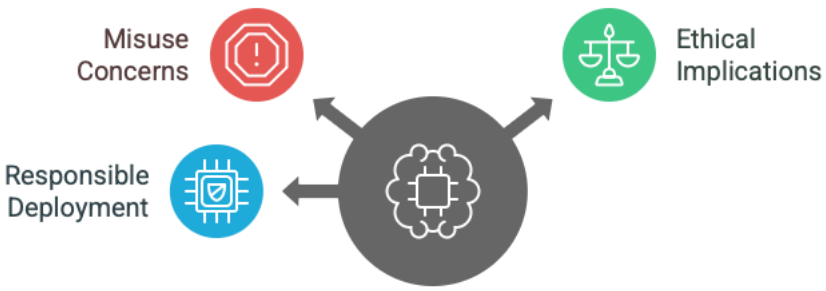


Figure 12. Limiting LLM Capabilities (NapkinAI 2025)

4.3.1 Setting Boundaries for Large Language Model Responses

One of the important dimensions to making large language models safe and acceptable in their response is limiting the response (Esmradi et al. 2024). The developers can impose such constraints on the maximum number of characters in the responses, the stylistic manner of the response, or which subject is excluded, making their outputs predictable and harmless (Zhao et al. 2023). There are a topics restriction prohibits to the large language model from responding to queries and that could potentially lead to leaking personal information or give harmful instruction (Ishibashi & Shimodaira 2023). Template answers form part of the alternative technique for standardizing outputs, providing harder grounds for the manipulation of the uninvited malicious command structures to the answer (Pingua et al. 2024). Indeed, boundaries can also be set through an adjustment in the decoding technique, whereas incorporating filters validate the generated text against a safety checklist before delivery (Zhang et al. 2025). Thus, even when hostile input is present, the model remains within secure operational limits (Esmradi et al. 2024).

To tweak some of these boundaries independently, developers usually apply dynamic adjustment techniques that modify the limits according to the context (Zhao et al. 2024). The real time analysis of both user inputs and model outputs is then used for detecting any deviations from the safe zone defined (Ishibashi & Shimodaira 2023). Once an out of normal pattern is detected, then the system can automatically trigger fallback response or escalate the incident for manual review (Pingua et al. 2024). Such adaptive boundary setting availing to application prompts minimal prompt-injection attack risk as well as enhanced overall robustness of application (Zhang et al. 2025). Thus, it is well agreed that sharp boundary establishment is one of the bases in defense strategy against LLM integrated systems (Esmradi et al. 2024).

4.3.2 Use of Safe Completion Policies

The Safe Completion Policies have a direct bearing on whether or not LLMs generate unsafe responses that cannot be trusted by the system and its users (Esmradi et al. 2024). Such trajectories provide LLMs with additional means to complete prompts in a safe manner, effectively substituting an objectionable output with a safe answer like "I don't know" (Zhao et al. 2023). Such measures work

well where prompts are being manipulated to coax sensitive or offensive outputs from the model (Ishibashi & Shimodaira 2023). Within a controlled framework of output generation, safe completion policies prevent a fair degree of unintended disclosures or granting of malicious commands (Pingua et al. 2024). Acquisition of safety measures can also bring these measures directly into play in the decoding process, where therefore every generated response will be cross checked against some criteria for safety before it gets to the user (Zhang et al. 2025).

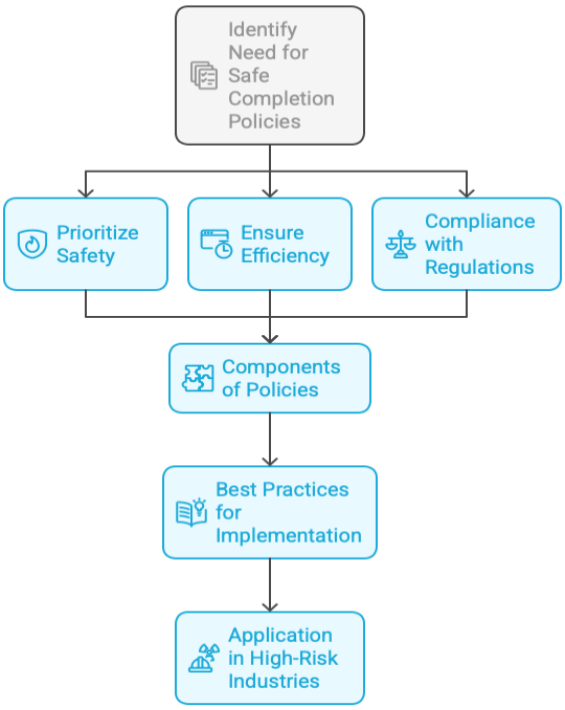


Figure 13. Safe Completion Policies Implementation (NapkinAI 2025)

Additionally, safe completion procedures can potentially be fine-tuned via reinforcement learning methods to further weight the scale toward safety and accuracy of replies (Esmradi et al. 2024). The fine-tuning involves training the LLM on a curated dataset that promotes safe and ethical outputs to strengthen them against adversarial manipulations (Zhao et al. 2023). Refining the policies based on the changing patterns in threats enables the developers to keep the model adjusted for new attack types without affecting its performance in the benign context (Ishibashi & Shimodaira 2023). So, in the absence of safe completion policies, models will be unable to keep any secure interaction environment in LLM powered applications, especially in high risk areas (Pingua et al. 2024).

4.4 Monitoring and Logging

Prompt injection attacks can mostly be countered with a very robust monitoring and logging system. Continuous monitoring is also necessary to look into the different variances in the types of inputs as well as the tokens used in order to detect some of the most peculiar things with little time taken (Pingua et al. 2024.) Logging every interaction would allow tracing every transaction to its source in case of malfunctioning abnormal conditions (Agarwal, Fabbri & Risher 2024). This sort of documentation would be important because it would show abrupt changes into normal patterns of input or longer than normal delays in returning to the action phase (Pasquini, Kornaropoulos & Ateniese 2024). Automated systems could work better with humans monitoring them to help detect a potential security breach much earlier (Pingua et al. 2024).

Figure 14. Security Monitoring and Logging (NapkinAI 2025)

Apart from this, it provides the chance to review these handles for more comprehensive incident investigations (Agarwal et al. 2024). The systematic, such as monitoring itself, makes for a much more reproducible answer when ascertaining the continued reliability and security of LLM-based systems (Pasquini et al. 2024).

4.4.1 Detecting Suspicious Activities

Detecting suspicious activities is a vital part of securing LLM-integrated applications against prompt

injection attacks (Esmradi et al. 2024). This requires an advanced monitoring system with the capability of in-depth analysis of user inputs and outputs generated for any anomalous activities that may signify adversarial movement (Zhao et al. 2023). Real-time detection algorithms would equip monitor tools that attempt any unusual pattern identification, such as an unexpected alteration in prompt structure or content that diverged from normal expectations (Ishibashi & Shimodaira 2023). The operation of these systems involved the use of machine learning techniques, whereby incoming data would be quickly measured and thereby compared against a good baseline generated from legitimate user interaction of sorts so that anything suspicious could be flagged (Pingua et al. 2024). Whenever suspicious activity is detected, alerts are sent for investigation, setting up a window to intercept potential escalation of an attack (Zhang et al. 2025). In addition, continuous logging of interactions aids in further investigation of incidents and on the job improvement of detection algorithms (Esmradi et al. 2024).

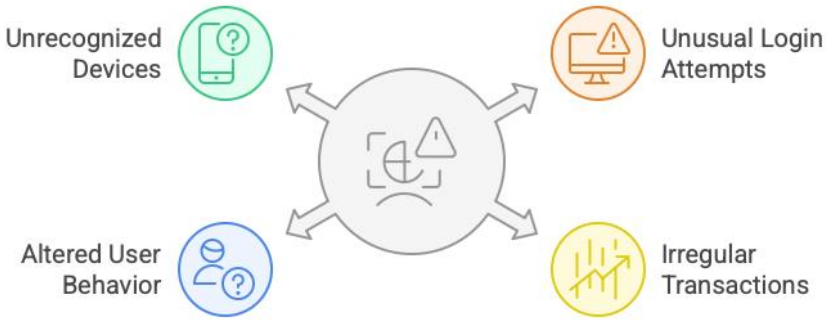


Figure 15. Detecting Suspicious Activities (NapkinAI 2025)

Through the actions undertaken by using such detection systems, organizations can shorten substantially the time window available to attackers so that intervention can take place before any extensive damage is inflicted (Zhao et al. 2023). The added advantage of anomaly detection as an extra measure of protection means a layered defense is achieved resulting in increased resilience of the system (Ishibashi & Shimodaira 2023). This kind of monitoring is especially important in the security of web-based LLM applications since methods of attack are constantly evolving (Pingua et al. 2024). The final impact of strong detection methods is to make sure any behavior diverging from the expected one is granted immediate attention, thus safeguarding the system itself and the confidence of its users (Zhang et al. 2025)

4.4.2 Incident Response Planning

Incident response planning acts as a final work-up defense line against prompt injection attacks and adversarial threats targeting LLM-embedded systems (Esmradi et al. 2024). Response plans quickly allow an organization to identify the anomaly, isolate affected systems, and eliminate damages emanating from security issues. With a robust response plan in place, it begins with proper attack vector identification, the involved breaches, and immediate isolation of affected areas of the system (Ishibashi et al. 2023). Further incident response protocols should have elaborate procedures for logging all interactions and keeping evidence for forensic examination (Pingua et al. 2024). This kind of guidance supports the investigation of incidents and also helps enhance the security measures (Zhang et al. 2025).

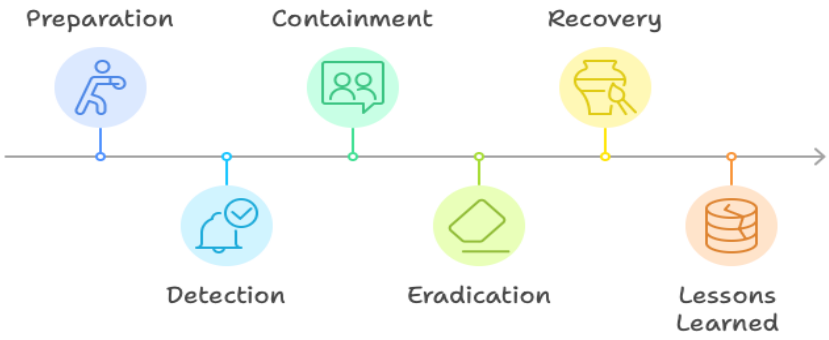


Figure 16. Incident Response Plan for LLM Applications (NapkinAI 2025)

Besides, exercising and simulations are the backbone to test incident response plan effectiveness, while also ensuring all stakeholders can respond promptly during an actual attack (Esmradi et al. 2024). The ability of an organization to respond to various scenarios of attacks depends on whether it has manual/automated responses set up to ensure remediation and containment of any breaches in a timely fashion (Zhao et al. 2023). The proactive nature ensures protection and resilience for a security framework, which itself needs constant adaptation to effectively deal with emerging threats (Ishibashi & Shimodaira 2023). The stronger the argument that a solid incident response plan can minimize

downtimes and, therefore, protect sensitive data and user trust for LLM integrated applications (Pingua et al. 2024).

4.5 Security Best Practices for Developers

Developers should secure their applications by designing in the phase wise manner, so as to protect them from injection attacks (Yao et al. 2024). Describe secure prompt engineering practices such as cleaning and validating user inputs that could significantly mitigate vulnerabilities.Incorporate verifications concerning the context and integrity of user inputs, reducing risks even more (Agarwal et al. 2024). Enhancing security would significantly improve with multiple layers of defenses such as multi-step validations and filtering techniques (Pingua et al. 2024). These methods resembled and are related to existing cybersecurity interventions that act against conventional injection attacks (Yao et al. 2024). Additional strategies include filtering of dangerous characters, where these methods use structured data format within the proof; thus, that in itself would add very much to the other defenses put in place to counter adversarial inputs (Liu et al. 2023).

The developers need to continuously secure their built applications by methods like code reviews, secure coding practices, and understanding what new threats evolve too (Pingua et al. 2024). These regular training sessions keep them up with the newest security trends and defenses against prompt injection attacks (Pasquini et al. 2024). These strategies for mitigation engender a multi-layered security architecture which both ends up protecting applications while maintaining user trust (Agarwal et al. 2024). The next section shall examine some of the real-world case studies to which these security practices apply (Pasquini et al. 2024).

5 CASE STUDIES

In this chapter, we provide case studies that describe how prompt injection attacks have really been hacked by artificial intelligence assistants on the web, how those incidents have affected and what lessons could be learned from them (Mathew 2024). Such examples are important as they recount how real-life attackers used LLM-integrated applications and showed fragile parts that compromise system integrity (Liu et al. 2023). The threat landscape can be better understood and the effectiveness of existing mitigation strategies could be evaluated-from actual incidents and their outcomes (Esmradi et al. 2024). The delivery brings together an incident overview and investigation of attack consequences, as well as conclusions of lessons learned for developing future defenses (Aditya, Chawla & Dhingra 2024). This framework not only stresses the need for security, but also for continuous security and rapid reaction in any securing AI systems (Naito, Watanabe & Mitsunaga 2023). Each use case adds weight to the case for strong security and further adds to the impetus for research into advanced mitigation techniques (Mathew 2024).

5.1 Case Study : Exploitation in Web-based Artificial Intelligence Assistants

The first case study involves the incident where an AI assistant deployed on a web platform frequented by users was compromised via a prompt injection attack (Liu et al. 2023). Here, an attacker, in a sort of stealth mode, appended the malicious command into a normal user query in order to make the assistant spit out unintended information that would compromise both user data and hurting internal system prompts (Mathew 2024). The attacker's injection went through the safety filters of the application and showed how easily an attacker could ensnare even the more established AI assistants that would rely solely on static prompt structures (Esmradi et al. 2024). While the abnormal outputs were detected by the logging mechanism of the system, the exploit came to light as part of routine monitoring (Aditya et al. 2024). It was a combination of an excessive case of input validation and an undue trust in the LLMs to interpret complex prompts accurately that made this exploitation happen (Naito et al. 2023). The case shows that even a small flaw in prompt handling may open the door for big wrongs (Mathew 2024). This incident has now led to a critical review and overhaul of the security measures in deployment on AI integrated web services (Liu et al. 2023).

5.1.1 Incident Overview

In the current case study, we have a subject very relevant to a well-known web-based AI assistant that was recently misused through prompt injection attack (Wu, Zhang & Jha 2024). The attacker appended harmful commands to the user input, thus hacking the original intended instruction causing unintended outputs produced by the AI assistant (Suo 2024). The said incident occurred while the system was operating normally when the attacker injected a command to circumvent standard safety filters and secretly revealed sensitive internal prompts (Kim, Song & Na 2024). In the investigation's findings, it has been revealed that the malevolent prompt could directly merge inside the legitimate user query to supersede the security constriction that provided safety (Hadi, Qureshi & Shah 2023). System logs showed that the attack took advantage of a weak input sanitization that allowed persistent commands to survive through processing multiple stages (Wu et al. 2024). Thus, the forensic analysis by the security team was prompted only after users reported some suspicious behavior of AI responses (Suo 2024). Early evaluations leaned toward a serious failure of distinguishing malicious from benign input (Kim et al. 2024). In this regard, the incident highlights the fact that even technically robust AI assistants can be susceptible to prompt injection techniques (Hadi et al. 2023).

External plugins and unverified web tools further risk the architecture of the system, as was found in the later examination of the system architecture (Wu et al. 2024). Attackers exploited the core flexibility within the prompt processing of the LLM which ignored robust validation of user inputs (Suo 2024). This failure let in commands that injected themselves without difficulty and outlines the need for comprehensive reviews against security while deploying an AI system (Kim et al. 2024). Moreover, it showed how easy it is for an attacker to manipulate contextual clues within the prompt to automatically effect a sequence of harmful operations (Hadi et al. 2023). Thus, this case would send a warning signal to the developers to recheck the robustness of their input handling mechanisms (Wu et al. 2024). It thus also becomes the event showing one possible failure in the flexibility-security balance of all applications interfacing with LLMs (Suo 2024).

5.1.2 Consequences of the Attack

The Prompt injection attack consequences were extensive and spanned both technical infrastructure and the user community (Wu et al. 2024). At a technical level, unauthorized injection leakage of internal system prompts along with sensitive operational data could be exploited for further

exploitation by the enemies in waiting (Kim et al. 2024). The breach not only shattered the sanity of the AI assistant but also disrupted its ordinary function to produce results that misled the users (Hadi et al. 2023). Financially, the provision of emergency security measures and loss of reputation value opened up financial avenues to potential losses that the service provider would incur as a result of the incident (Suo et 2024). On top of that, users complained that the AI assistant's response began to deviate from that intended by its internal processes, creating confusion and undercutting the system's credibility (Wu et al. 2024). In addition to immediate technical repercussions, the break was long-term in its injury to user confidence, especially with regard to enterprise clients using the system to conduct critical operations (Kim et al. 2024). Regulatory concerns were also raised about breaches in the vulnerability with regard to compliance in data privacy and security (Hadi et al. 2023). The attack thus triggered a wide-ranging examination of security protocols across similar web-based LLM applications (Suo 2024).

Figure 17. Consequences of Attacking LLMs (NapkinAI 2025)

Moreover, the exploitation showed that the used vulnerability was not peculiar to this application but might be adaptable to many other applications of this kind (Wu et al. 2024). The expansive nature of the vulnerability raised a large alarm among industry stakeholders that immediately began calling for urgent standardization of security measures to prevent potential occurrences of the danger (Kim et al. 2024). Success by the attacker to pull out sensitive data further emphasized the necessity of implementing very strong defense mechanisms to protect end-user privacy (Hadi et al. 2023). Operational, financial, and reputational costs together showed that outcomes of prompt injection attacks do not limit themselves to overcoming technical breaches, but they cut across the whole ecosystem within which AI-based services work (Suo 2024). These multifaceted effects have initiated

a debate among researchers concerning the trade-offs between flexibility in the model as well as security, demanding a reconsideration of the existing safety assumption (Wu et al. 2024).

5.1.3 Lessions Learned

This incident has given lessons that are important for security practices in LLM integrated applications in the future (Wu et al. 2024). First, it underscored the absolute requirement for extensive input validation and sanitization to ensure that the system processes only trusted commands (Suo 2024). Developers would be required to adopt multi-layered security approaches that take both preventive and detective controls in order to mitigate risk posed by prompt injection attacks (Kim et al. 2024). Additionally, the system is severely curtailing robust logging and monitoring systems, with early detection of anomalies being much useful in reducing the impact of attacks (Hadi et al. 2023). The incident also brought out the case that it is not only important that the whole LLM ecosystem should be assessed for security but that the core language model is captured as well (Wu et al. 2024). This broader perspective is very much important because it can unearth vulnerabilities brought about by additional components that the development invests in, such as plugins and other web tools and interfaces for users (Suo 2024). Along the same lines, ongoing training and awareness for developers towards new attack techniques become imperative to preempt new vulnerabilities in future development (Kim et al. 2024).

Finally, it became clear that one could invest massively in research on learning how to counter with advanced defense mechanisms, such as signed prompts and dynamic context verification, against the kinds of attacks in the future (Hadi et al. 2023). Thus, from this case study, proactive security measures are, indeed, vital to mitigating risks involving prompt injection from LLM-integrated applications (Wu et al. 2024). This show how proactive measures learned guide a radically different system-wide perspective of security, which should include robust input sanitization and effective monitoring coupled with continuous updates to defense strategies (Suo 2024). Such knowledge would not only enrich the design of more secure systems, but would also lay the foundation for future research into advanced mitigation techniques that are appropriate to the dynamic nature of AI-based applications (Kim et al. 2024).

6 DISCUSSION

This chapter discusses issues relating to the challenges, effectiveness, and future mitigation of prompt injection attacks in web-based LLM applications (Wu et al. 2024). We synthesize the findings from the case studies and broader literature to identify key obstacles towards preventing these attacks and to evaluate the current status of mitigation strategies (Suo 2024). This inquiry will provide a cohesive understanding of the underlying challenges and suggest a direction towards securing LLM-integrated systems (Kim et al. 2024). The discussion is divided into three main sections like challenges in prevention, effectiveness of certain mitigation strategies, and future considerations for the development of secure web-based LLM applications (Hadi et al. 2023).

6.1 Challenges in Preventing Prompt Injection Attacks

Natural language's inherent flexibility is one of the biggest challenges in prompt injection attack prevention, allowing crafted inputs that are difficult to filter by any traditional security weaponry (Wu et al. 2024). The dynamic and context-dependent nature of LLMs translates as failing detection due to minor changes in prompt phrasing; thus their static filtering techniques hardly come close to addressing such vulnerabilities (Suo 2024). The merging of several components such as web tools and external plugins creates interaction surfaces complicated enough to provide an easy injection source for attackers (Kim et al. 2024). Another challenge to be faced is the trade-off between flexibility and security of a model; very strict filters would be at the expense of performance and user experience, while lenient filter applications present the system to higher risk exposure (Hadi et al. 2023).

The constant evolution of attacker tactics adds another layer of complexity in developing efficient countermeasures, as the defense strategies can always transform to exploit newly introduced techniques of exploitation (Wu et al. 2024). Also adding on top of this is the extremely resource-hungry continuous update and monitoring of the systems, which would pose practical challenges to developers (Suo 2024). Finally, the absence of standard evaluation metrics for security in LLM applications makes it rather difficult to compare the effectiveness of the different mitigation strategies (Kim et al. 2024). Overall, prompt injection attacks threaten to undermine the delicate balance of security and usability and continuous research and development to counter emerging threats (Hadi et al. 2023).

Figure 18. Challenges in LLM Security (NapkinAI 2025)

The probabilistic output of these LLMs, furthermore, added to the complexities surrounding detection, benign prompts are sometimes known to spout unexpected outputs (Wu et al. 2024). It is this uncertainty that impairs the reliability of automated defense systems that rely on consistent behavior from the LLM (Suo 2024). In addition, since LLM architectures are still continuously evolving, many issues may come with their generation, requiring a frequent re-evaluation and updating of defense strategies (Kim et al. 2024). The matter is compounded by the more black-boxed approaches taken by most commercial LLMs, where the internal workings of the LLMs are not fully explained, hampering the assessment (Hadi et al. 2023). To provide a high level of security for those systems, therefore, one has to have a multi-faceted approach involving advanced monitoring, anomaly detection, and adaptive countermeasures (Wu et al. 2024).

6.2 Effectiveness of Basic Mitigation Strategies

There have been some successes with basic mitigation strategies such as input validation and sanitization, user authentication, and role-based access control to dampen the risk for prompt injection attacks (Wu et al. 2024). Input validation dissects user inputs to pass on only the benign commands to LLM, hence forming a primary layer of defense (Suo 2024). But the claim gets worn down due to other constraints such as the very complexity and variable nature of human speech, rendering all of its

inputs as possible false negatives, thus letting some harmful inputs through (Kim et al. 2024). Similarly, user authentication and strong access control create problems for unauthorized personnel trying to insert a harmful prompt into the system, however, the potential risk of insider accounts or compromised accounts still exists in this case (Hadi et al. 2023). The trade-off between security and usability makes the challenge even more difficult, as excessive restrictions can impede valid interactions (Naito et al. 2023).

Techniques for curbing LLMs, like putting limits on the model response or following "safe completion" pathways, have proven beneficial by suppressing harmful outputs, but still could impair some functionality of the system and thus lower its utility altogether (Wu et al. 2024). Besides, monitoring and logging allow real-time detection of any abnormal behavior, which would allow for a quick response to emerging threats, although this approach is resource-intensive and continuously needs tuning to keep itself relevant (Suo 2024). In conclusion, while basic mitigation strategies may form a good foundation upon which to defend against these risks, they are not perfect and need to be fit into larger multi-layered security architecture (Kim et al. 2024).

In practice, the effectiveness of any combination would implement several mitigation strategies, forming a stronger defense against prompt injection attacks (Hadi et al. 2023). For example, the use of input validation combined with dynamic output filtering and real-time monitoring creates an overlapping layer of security that catches and mitigates attacks at different stages of the data processing pipeline (Wu et al. 2024). Regardless, the success of these measures could be thwarted by the fast-paced advancement of adversarial approaches, usually modifying themselves to sidestep single defenses (Suo 2024). Hence, one must regularly assess and improve mitigation techniques to achieve an appropriate level of security (Kim et al. 2024). One could therefore clearly state that, while existing strategies can provide limited protection, they will need to continue to evolve in order to address emerging vulnerabilities and thus secure the deployment of LLM-based systems (Hadi et al. 2023).

6.3 Future Considerations for Web-Based Large Language Model Application

In general, web-based LLM applications' future security depends on the evolution of more adaptive and proactive defense systems (Wu et al. 2024). In addition to future research focusing on context-aware filtering and anomaly detection algorithms to fuse highly automated and dynamic responses to novel attack vectors, more advanced threat detection and prevention methods would be included (Suo

2024). More extensive mitigation techniques would also require development, including stronger mechanisms for prompt authentication, user authentication, and signed prompts, to prevent prompt injection attacks (Kim et al. 2024). Setting up a systematic security framework and evaluation metrics will be imperative as LLM standards are advancing and integrating deeper into the variety of web-based applications for proper protection of the various components of the system (Hadi et al. 2023). In addition, a decentralized security form in which distributed monitoring and collaborative defense mechanisms contribute would give a more resilient safeguard against attacks (Wu et al. 2024). It is equally important to note that industrial players should collaborate closely with academic researchers on threat intelligence and good practices in securing LLM-integrated applications (Suo 2024).

The new trends within the implementation of LLMs on the web would thus require the establishment of standardized security frameworks and the evaluation metrics necessary for each component of the system to be adequately protected (Hadi et al. 2023). Further, one possible approach could include decentralized security forms, which have distributed monitoring and collaborative defense mechanisms, thereby providing more resilient security from attacks (Wu et al. 2024). Such partnerships allow academic researchers and industry stakeholders to share threat intelligence and best practices in securing LLM-integrated applications(Suo 2024). Changing history such as is above makes it inauspicious within which it happens.

The protection of web-based LLM applications in future scenarios is going to largely depend upon the progress of defense mechanisms that are more adaptive and proactive (Wu et al. 2024). Future research should majorly focus on integrating advanced threat detection methods, such as context-aware filtering and anomaly detection algorithms, which can dynamically respond to new attack vectors (Suo 2024). Moreover, the study of further advanced mitigation techniques, including but not limited to the above, signed prompts, and more robust protocols for user authentication, is essential to prevent prompt injection attacks (Kim et al. 2024). It would also emerge that, as LLMs advance further and deeper into a variety of web-based applications, standardized frameworks and evaluation metrics would set the stage for their requisite security (Hadi et al. 2023). In the future of safety plan is expected to be decentralized, such that both monitoring and defense mechanisms are distributed, and hence, a more firewall against any attacks would be more effective (Wu et al. 2024). To the extent that partnerships could also potentially be leveraged into a contribution towards collaborative threat intelligence or even best practices for securing the LLMs to enabled between academia and industry players (Suo 2024). The logical set of standards in the web-based LLM application implementations would require framing standard security frameworks and evaluation metrics that each system component must be adequately

protected with (Hadi et al. 2023). The security such as can also take a decentralized form that will include distributed monitoring and collaborative defense mechanisms that can make the safety resilient against from the attacks (Wu et al. 2024). However, in this partnerships to enables sharing among academic researchers and industry players of threat intelligence and with best practices in securing the LLM-integrated applications (Suo 2024).

Considering future scenarios, the protection of web-based LLM applications is significantly dependent on the advancement of defense mechanisms more adaptive and proactive (Wu et al. 2024). Future research must stimulate itself on advanced threat detection methods such as integration of context-aware filtering and anomaly detection algorithms that can automatically respond dynamically to new attack vectors (Suo 2024). More research is also necessary on other advanced mitigation techniques including but not limited to signed prompts and stronger protocols for user authentication which are essential against prompt injection attacks (Kim et al. 2024). Further, it would be found that as LLM continues advancing deeper and further into different web applications, a need would arise for standardized frameworks and evaluation metrics to have the required level of security for all components of the system (Hadi et al. 2023). The addition of decentralized security forms in which distributed monitoring and collaborative defense mechanisms contribute would give a more resilient safeguard against attacks (Wu et al. 2024). Also, it is essential that industry players engage deeply with academic researchers in the sharing of intelligence regarding threats and best practices in the securing of LLM-integrated applications (Suo 2024).

Lastly, it would be preferred that future systems come with continuous learning and adaptations to recognize and fight new threats while sacrificing experience as little as possible (Kim et al. 2024). It should also aim for developing AI-based security solutions that establish not only the detection of prompt injection attacks but also change system parameters automatically to nullify their effect online (Hadi et al. 2023). As more services online emerge with LLMs in critical applications, regulatory frameworks may in the future demand more stringent governance in security and audit requirements, thus further underscoring the need for robust, flexible defenses (Wu et al. 2024). Most importantly, the long-term sustainability and trustworthiness of these LLM-based applications will depend on ensuring that technological development and security are kept in balance (Suo 2024).

7 CONCLUSION

In conclusion, this thesis probed the intricacies of the problem of prompt injection attacks in web-based large language model (LLM) applications and suggested sets of elementary countermeasures that can bolster the security of systems. We have shown in this study that prompt injection attacks far more seriously compromise the integrity of the AI-powered systems by manipulating the intended functionality of LLMs and therefore eroding user trust. Real-world case studies were analyzed by us showing how adversaries exploit the vulnerabilities of prompt structures, which emphasizes the dire need for setting up stronger defensive measures.

In the course of our research, we found that even if current mitigation strategies are providing a good basis, they alone will not be able to second all kinds of high-level attacks. This illustrates the continuing problem of balancing security with just the functional flexibility that makes LLMs so worthwhile. It is suggested through the results of our investigations that more adaptive and multi-layered approaches are warranted, where the focus will be on real-time response to changes in threat landscapes.

This thesis can help further studies in improving the resiliency and reliability of AI-oriented applications. On the understanding of the subtleties of these vulnerabilities, both developers and researchers can synthesize for building far more secure, efficient systems that could save further sensitive data while performing high on such activities. Undeniably, the safe deployment of LLMs will prove vital as AI systems take on an even more central role in digitalized solutions and services.

REFERENCES

Abdali, S., Anarfi, R., Barberan, C.J. and He, J., 2024. Securing large language models: Threats, vulnerabilities and responsible practices. arXiv preprint arXiv:2403.12503.Available at: https://doi.org/10.48550/arXiv.2403.12503. Accessed 19 Februray 2025.

Aditya, H., Chawla, S., Dhingra, G., Rai, P., Sood, S., Singh, T., Wase, Z.M., Bahga, A. and Madisetti, V.K., 2024. Evaluating privacy leakage and memorization attacks on large language models (llms) in generative ai applications. Journal of Software Engineering and Applications, 17(5), pp.421-447.Available at: doi:10.4236/jsea.2024.175023 . Accessed 21 Januray 2025.

Agarwal, D., Fabbri, A.R., Risher, B., Laban, P., Joty, S. and Wu, C.S., 2024. Prompt Leakage effect and defense strategies for multi-turn LLM interactions. arXiv preprint arXiv:2404.16251. Available at: https://10.18653/v1/2024.emnlp-industry.94. Accessed 19 Januray 2025.

Aryal, K., Gupta, M., Abdelsalam, M. and Saleh, M., 2024. Intra-section code cave injection for adversarial evasion attacks on windows pe malware file. arXiv preprint arXiv:2403.06428. Available at: https://doi.org/10.48550/arXiv.2403.06428. Accessed 19 Februray 2025.

Berti, L., Giorgi, F. and Kasneci, G., 2025. Emergent Abilities in Large Language Models: A Survey. arXiv preprint arXiv:2503.05788.Avaiable at: https://doi.org/10.48550/arXiv.2503.05788. Accessed 19 Februray 2025.

Bolón-Canedo, V., Morán-Fernández, L., Cancela, B. and Alonso-Betanzos, A., 2024. A review of green artificial intelligence: Towards a more sustainable future. Neurocomputing, p.128096. Available at: https://doi.org/10.1016/j.neucom.2024.128096. Accessed 30 March 2025.

Bubeck, S., Chadrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y.T., Li, Y., Lundberg, S. and Nori, H., 2023. Sparks of artificial general intelligence: Early experiments with gpt-4 [online]. Available at: https://doi.org/10.48550/arXiv.2303.12712. Accessed 21 Februray 2025.

Divakaran, D.M. and Peddinti, S.T., 2024. LLMs for cyber security: New opportunities. arXiv preprint arXiv:2404.11338. Available at: https://doi.org/10.48550/arXiv.2404.11338. Accessed 23 Februray 2025.

Dong, Y., Mu, R., Jin, G., Qi, Y., Hu, J., Zhao, X., Meng, J., Ruan, W. and Huang, X., 2024. Building guardrails for large language models. arXiv preprint arXiv:2402.01822. Available at: https://doi.org/10.48550/arXiv.2402.01822. Accessed 19 Februray 2025

Esmradi, A., Yip, D.W. and Chan, C.F., 2023, November. A comprehensive survey of attack techniques, implementation, and mitigation strategies in large language models. In International Conference on Ubiquitous Security (pp. 76-95). Singapore: Springer Nature Singapore. Available at: https://doi.org/10.1007/978-981-97-1274-86. Accessed 30 Februray 2025.

Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T. and Fritz, M., 2023, November. Not what you've signed up for: Compromising real-world llm-integrated applications with

indirect prompt injection. In Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security (pp. 79-90). Available at: https://doi.org/10.1145/3605764.3623985. Accessed 30 Februray 2025.

Hadi, M.U., Qureshi, R., Shah, A., Irfan, M., Zafar, A., Shaikh, M.B., Akhtar, N., Wu, J. and Mirjalili, S., 2023. A survey on large language models: Applications, challenges, limitations, and practical usage. Authorea Preprints, 3. Available at: doi: https://10.36227/techrxiv.23589741.v3. Accessed 21 Februray 2025.

Hackett, W., Birch, L., Trawicki, S., Suri, N. and Garraghan, P., 2025. Bypassing Prompt Injection and Jailbreak Detection in LLM Guardrails. arXiv preprint arXiv:2504.11168. Available at: https://doi.org/10.48550/arXiv.2504.11168. Accessed 19 march 2025.

Hao, Y., Yang, W. and Lin, Y., 2024. Exploring backdoor vulnerabilities of chat models. arXiv preprint arXiv:2404.02406.Available at: https://doi.org/10.48550/arXiv.2404.02406 . Accessed 30 Februray 2025.

Hau, K., Hassan, S. and Zhou, S., 2025. LLMs in Mobile Apps: Practices, Challenges, and Opportunities. arXiv preprint arXiv:2502.15908. Available at: https://doi.org/10.48550/arXiv.2502.15908. Accessed 19 march 2025.

He, Y., Qiu, J., Zhang, W. and Yuan, Z., 2024. Fortifying Ethical Boundaries in AI: Advanced Strategies for Enhancing Security in Large Language Models. arXiv preprint arXiv:2402.01725. Available at: https://doi.org/10.48550/arXiv.2402.01725. Accessed 21 March 2025.

Huang, Y., Zhan, R., Wong, D.F., Chao, L.S. and Tao, A., 2025. Intrinsic Model Weaknesses: How Priming Attacks Unveil Vulnerabilities in Large Language Models. arXiv preprint arXiv:2502.16491.Available at: https://doi.org/10.48550/arXiv.2502.16491. Accessed 19 March 2025.

Ishibashi, Y. and Shimodaira, H., 2023. Knowledge sanitization of large language models. arXiv preprint arXiv:2309.11852. Available at: https://doi.org/10.48550/arXiv.2309.11852. Accessed 21 Februray 2025

Khomsky, D., Maloyan, N. and Nutfullin, B., 2024, September. Prompt injection attacks in defended systems. In International Conference on Distributed Computer and Communication Networks (pp. 404-416). Cham: Springer Nature Switzerland. Available at: https://doi.org/10.48550/arXiv.2406.14048. Accessed 19 Februray 2025.

Kim, H., Song, M., Na, S.H., Shin, S. and Lee, K., 2024. When LLMs Go Online: The Emerging Threat of Web-Enabled LLMs. arXiv preprint arXiv:2410.14569. Available at: https://doi.org/10.48550/arXiv.2410.14569. Accessed 13 Februray 2025.

Kumarage, T., Bhattacharjee, A. and Garland, J., 2024. Harnessing artificial intelligence to combat online hate: Exploring the challenges and opportunities of large language models in hate speech detection. arXiv preprint arXiv:2403.08035. Available at: https://doi.org/10.48550/arXiv.2403.08035. Accessed 19 march 2025.

Lee, D., Whang, T., Lee, C. and Lim, H., 2023. Towards reliable and fluent large language models: Incorporating feedback learning loops in qa systems. arXiv preprint arXiv:2309.06384.Available at: https://doi.org/10.48550/arXiv.2309.06384. Accessed 19 Februray 2025.

Liu, A., Zhou, Y., Liu, X., Zhang, T., Liang, S., Wang, J., Pu, Y., Li, T., Zhang, J., Zhou, W. and Guo, Q., 2024. Compromising embodied agents with contextual backdoor attacks. arXiv preprint arXiv:2408.02882. Available at: https://doi.org/10.48550/arXiv.2408.02882. Available at: Accessed 19 Februray 2025.

Liu, T., Deng, Z., Meng, G., Li, Y. and Chen, K., 2024, December. Demystifying rce vulnerabilities in llm-integrated apps. In Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security (pp. 1716-1730). Available at: https://doi.org/10.1145/3658644.3690338. Accessed 19 Februray 2025.

Liu, Y., Deng, G., Li, Y., Wang, K., Wang, Z., Wang, X., Zhang, T., Liu, Y., Wang, H., Zheng, Y. and Liu, Y., 2023. Prompt Injection attack against LLM-integrated Applications. arXiv preprint arXiv:2306.05499. Available at: https://doi.org/10.48550/arXiv.2306.05499. Accessed 19 March 2025

Liu, Y., Jia, Y., Geng, R., Jia, J. and Gong, N.Z., Prompt Injection Attacks and Defenses in LLM-Integrated Appli-cations, October 2023c. Available at: https://doi.org/10.48550/arXiv.2310.12815. Accessed 19 March 2025

Liu, Z., Tang, L., Sun, Z., Liu, Z., Lyu, Y., Ruan, W., Xu, Y., Shan, L., Shin, J., Chen, X. and Zhu, D., 2025. AD-GPT: Large Language Models in Alzheimer's Disease. arXiv preprint arXiv:2504.03071. Available at: https://doi.org/10.48550/arXiv.2504.03071. Accessed 30 march 2025.

Madani, S., Tavasoli, A., Astaneh, Z.K. and Pineau, P.O., 2025. Large Language Models integration in Smart Grids. arXiv preprint arXiv:2504.09059.Available at: https://doi.org/10.48550/arXiv.2504.09059. Accessed 19 Februray 2025.

Mathew, E., 2024. Enhancing Security in Large Language Models: A Comprehensive Review of Prompt Injection Attacks and Defenses. Authorea Preprints.Available at: https://doi.org/10.36227/techrxiv.172954263.32914470/v1. Accessed 21 Januray 2025.

Minaee, S., Mikolov, T., Nikzad, N., Chenaghlu, M., Socher, R., Amatriain, X. and Gao, J., 2024. Large language models: A survey, 2024. arXiv preprint arXiv:2402.06196. Available at: https://doi.org/10.48550/arXiv.2402.06196. Accessed 13 Januray 2025.

Naito, T., Watanabe, R. and Mitsunaga, T., 2023, November. Llm-based attack scenarios generator with it asset management and vulnerability information. In 2023 6th International Conference on Signal Processing and Information Security (ICSPIS) (pp. 99-103). IEEE.Available at: 10.1109/ICSPIS60075.2023.10344019. Accessed 30 Januray 2025.

Naveed, H., Khan, A.U., Qiu, S., Saqib, M., Anwar, S., Usman, M., Akhtar, N., Barnes, N. and Mian, A., 2023. A comprehensive overview of large language models. arXiv preprint arXiv:2307.06435.https://doi.org/10.48550/arxiv.2307.06435. Accessed 13 Januray 2025.

Pan, J., Wong, S.L., Yuan, Y. and Chia, X.W., 2025. Prompt Inject Detection with Generative Explanation as an Investigative Tool. arXiv preprint arXiv:2502.11006. Available at: https://doi.org/10.48550/arXiv.2502.11006. Accessed 19 Februray 2025.

Pasquini, D., Kornaropoulos, E.M. and Ateniese, G., 2024. Hacking Back the AI-Hacker: Prompt Injection as a Defense Against LLM-driven Cyberattacks. arXiv preprint arXiv:2410.20911. Available at: https://doi.org/10.48550/arxiv.2410.20911. Accessed 30 Januray 2025. Accessed 19 Februray 2025.

Pantha, N., Ramasubramanian, M., Gurung, I., Maskey, M. and Ramachandran, R., 2024. Challenges in Guardrailing Large Language Models for Science. arXiv preprint arXiv:2411.08181. Available at : https://doi.org/10.48550/arXiv.2411.08181. Accessed 19 March 2025

Pedro, R., Castro, D., Carreira, P. and Santos, N., From prompt injections to SQL injection attacks: how protected is your LLM-integrated web application?(2023). arXiv preprint arXiv:2308.01990.Available at. https://doi.org/10.48550/arXiv.2308.01990. Accessed 19 March 2025.

Pingua, B., Murmu, D., Kandpal, M., Rautaray, J., Mishra, P., Barik, R.K. and Saikia, M.J., 2024. Mitigating adversarial manipulation in LLMs: a prompt-based approach to counter Jailbreak attacks (Prompt-G). PeerJ Computer Science, 10, p.e2374.Available at: https://doi.org/10.7717/peerj-cs.2374. Accessed 19 march 2025.

Qiang, Y., Zhou, X. and Zhu, D., 2023. Hijacking large language models via adversarial in-context learning. arXiv preprint arXiv:2311.09948. Available at: https://doi.org/10.48550/arXiv.2311.09948. Accessed 21 Februray 2025.

Rababah, B., Wu, S.T., Kwiatkowski, M., Leung, C.K. and Akcora, C.G., 2024, December. Sok: Prompt hacking of large language models. In 2024 IEEE International Conference on Big Data (BigData) (pp. 5392-5401). IEEE. Available at: doi: 10.1109/BigData62323.2024.10825103. Accessed 13 Februray 2025.

Raza, M., Jahangir, Z., Riaz, M.B., Saeed, M.J. and Sattar, M.A., 2025. Industrial applications of large language models. Scientific Reports, 15(1), p.13755.Available at: https://doi.org/10.1038/s41598-025-98483-1. Accessed 21 Februray 2025.

Suo, X., 2024, December. Signed-prompt: A new approach to prevent prompt injection attacks against llm-integrated applications. In AIP Conference Proceedings (Vol. 3194, No. 1). AIP Publishing. Available at: https://doi.org/10.1063/5.0222987. Accessed 19 February 2025

Sakib, M.N., Islam, M.A., Pathak, R. and Arifin, M.M., 2024, September. Risks, Causes, and Mitigations of Widespread Deployments of Large Language Models (LLMs): A Survey. In 2024 2nd International Conference on Artificial Intelligence, Blockchain, and Internet of Things (AIBThings) (pp. 1-7). IEEE.Available at http://doi.org/10.1109/AIBThings63359.2024.10863356. Accessed 21 Februray 2025.

Wu, F., Zhang, N., Jha, S., McDaniel, P. and Xiao, C., 2024. A new era in llm security: Exploring security concerns in real-world llm-based systems. arXiv preprint arXiv:2402.18649. https://doi.org/10.48550/arXiv.2402.18649. Accessed 13 February 2025.

Yamamura, A. and Ganguli, S., 2024. Fooling LLM graders into giving better grades through neural activity guided adversarial prompting. arXiv preprint arXiv:2412.15275. Available at: https://doi.org/10.48550/arXiv.2412.15275. Accessed 21 Februray 2025.

Yao, Y., Duan, J., Xu, K., Cai, Y., Sun, Z. and Zhang, Y., 2024. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. High-Confidence Computing, p.100211. Available at: https://doi.org/10.1016/j.hcc.2024.100211. Accessed 19 Januray 2025.

Yao, Y., Jin, H., Shah, A.D., Han, S., Hu, Z., Ran, Y., Stripelis, D., Xu, Z., Avestimehr, S. and He, C., 2024. ScaleLLM: A Resource-Frugal LLM Serving Framework by Optimizing End-to-End Efficiency. arXiv preprint arXiv:2408.00008. Available at: https://doi.org/10.48550/arXiv.2408.00008. Accessed 19 march 2025.

Zhan, Q., Fang, R., Panchal, H.S. and Kang, D., 2025. Adaptive Attacks Break Defenses Against Indirect Prompt Injection Attacks on LLM Agents. arXiv preprint arXiv:2503.00061.Available at: https://doi.org/10.48550/arXiv.2503.00061 Accessed 19 Februray 2025.

Zhang, R., Li, H.W., Qian, X.Y., Jiang, W.B. and Chen, H.X., 2025. On large language models safety, security, and privacy: A survey. Journal of Electronic Science and Technology, p.100301. Available at: https://doi.org/10.1016/j.jnlest.2025.100301. Accessed 21 Februray 2025. Accessed 19 march 2025.

Zhang, R., Sullivan, D., Jackson, K., Xie, P. and Chen, M., 2025. Defense against Prompt Injection Attacks via Mixture of Encodings. arXiv preprint arXiv:2504.07467. Available at: https://doi.org/10.48550/arXiv.2504.07467. Accessed 21 Februray 2025.

Zhao, W.X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z. and Du, Y., 2023. A survey of large language models. arXiv preprint arXiv:2303.18223, 1(2). Available at: https://doi.org/10.48550/arxiv.2303.18223.Accessed 19 march 2025.

Zhang, Y., Jin, H., Meng, D., Wang, J. and Tan, J., 2024. A comprehensive survey on process-oriented automatic text summarization with exploration of llm-based methods. arXiv preprint arXiv:2403.02901.Available at: https://doi.org/10.48550/arXiv.2403.02901. Accessed 19 Februray 2025.

Zhang, Z., Bai, F., Chen, Q., Ma, C., Wang, M., Sun, H., Zheng, Z. and Yang, Y., 2025. Amulet: Realignment during test time for personalized preference adaptation of LLMs. arXiv preprint arXiv:2502.19148. Available at: https://doi.org/10.48550/arXiv.2502.19148. Accessed 21 Februray 2025.