International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 10 | Oct 2026 www.irjet.net p-ISSN: 2395-0072

PROMPT INJECTION ATTACKS IN LLM-INTEGRATED APPLICATIONS

Prof. Poonam Surve[1], Priyanshu Sandeep Devadkar[2], Omkar Manoj Patil [3], Rutvik Ajit Kadam[4]

[1]Assistant professor, D.Y.Patil Deemed To Be University School Of Humanities & Sciences Navi Mumbai, Maharashtra, India.

[234]Student, Department of MCA, D.Y.Patil Deemed To Be University School Of Humanities & Sciences Navi Mumbai, Maharashtra, India.

Abstract - *Large Language Models (LLMs) are being increasingly leveraged by a wide variety of applications including customer support chatbots, Retrieval-Augmented Generation (RAG) systems, coding assistants and AI-based agents. While offering a range of benefits over traditional approaches these models add additional layers of complexity to the security landscape. Recent advances in the field of adversarial machine learning reveal that it is possible to mount prompt injection attacks on such models by feeding misleading or malicious instructions to them via user inputs or external content. This paper classifies the main types of prompt injection attacks, analyzing how they affect the functioning of LLM-integrated applications and what security measures are proposed in the literature. Earlier studies explored methods of indirect prompt injection through external content, potential vulnerabilities of LLM-integrated applications, and various ways of attacking such systems [1]–[4].The paper analyzes the methods of input filtering, isolation of trusted and untrusted content, output validation, implementation of security guardrails, and least-privilege access control. To validate the findings of the literature review, the paper conducts a survey among 60 students to evaluate awareness of prompt injection attacks, the frequency of use of such applications, concerns about safety, and opinions about the security measures.Since the study does not include an experimental part with quantitative measurements, no specific conclusions about the likelihood of attacks or the efficiency of the proposed security measures can be reported. Instead, the paper summarizes the results of the survey and the findings from the literature review. The research methodology involves a literary review of the technical research papers and questionnaires surveying the understanding of prompt injection threats and awareness levels of users of such LLM applications. Preliminary analysis of survey responses indicates that while most users do not have a deep technical understanding of such attacks, they are generally concerned about such threats and have reasonable perception regarding the usefulness of different countermeasures. While the survey is still ongoing (60 students have been surveyed so far), the paper will discuss statistical analysis (using descriptive statistics, frequencies and percentages) of the survey results if available. The paper will draw together the technical research literature and the survey results to highlight key insights and areas of focus for the research. Prompt injection attacks pose a significant security challenge to a wide variety of applications that are deploying LLMs including RAG systems and AI agents. This paper will analyze the technical aspects of such attacks and evaluate different approaches that can be employed to harden such systems against them.*

Keywords: Prompt Injection, LLM Security, RAG Security, AI Agents, Cybersecurity, Defensive Guardrails, Adversarial Machine Learning.

1. Introduction

Large Language Models (LLMs) have become an integral part of modern computer applications due to their ability to comprehend natural-language instructions, analyse information, and produce human-like responses. These models power a variety of modern applications, including AI chatbots, AI assistants, computer customer service, coding assistants, document analysis tools, and Retrieval-Augmented Generation (RAG) systems. As the use of LLMs increases, these technologies improve the efficiency of numerous applications and fundamentally change the way users interact with modern computers.

At the same time, applications that integrate LLMs often face new security challenges, one of which is represented by prompt injection attacks. In general, a prompt injection represents an attack technique that allows an adversary to affect the target LLM by injecting malicious instructions into information processed by the model. In doing so, an attacker can bypass the application's security mechanisms, retrieve restricted data, or compromise the integrity of generated results.

Compared to traditional cybersecurity threats, prompt injection attacks are a relative novelty that specifically targets applications that utilise LLMs. At the same time, such applications are increasingly common as businesses seek to improve customer service, among other quality improvements. In a typical scenario, an application that uses LLM processes some information, where malicious instructions can be injected in either direct or indirect ways. As a result, the model acts according to the attacker's wishes, violating the expected application logic.

In general, prompt injection attacks can be separated into direct and indirect attack vectors. As the name suggests,

direct prompt injection involves introducing or inserting malicious instructions directly into the application. For example, if an application allows users to enter natural language prompts, an attacker can input specially crafted instructions to obtain restricted data or compromise the application.

Meanwhile, in the case of indirect prompt injection, malicious instructions are incorporated into another object, document, or web page that is processed by an LLM. Therefore, if an application is not designed to prevent such data from being processed, an attacker can utilise specially crafted documents or web content to inject malicious prompts into the LLM. In practice, such attacks can only affect applications that utilise RAG or AI agents. The first type of applications can be compromised if an attacker modifies the contents of a document that is analysed by an LLM. In the case of AI agents, the problem stems from the fact that they often utilise various tools, including databases, external API services, files, and applications. As a result, a successful prompt injection can cause such an application to leak sensitive information or utilise functions that are not permitted under normal operation conditions.

Researchers have proposed several approaches to address the threat of prompt injection. These include injecting special tokens, training specialised models that can detect malicious prompts, altering the application context to separate trusted and untrusted content, implementing security guardrails, and utilising privilege access control to protect sensitive information. At the same time, security solutions always entail some sacrifices in terms of application quality and user experience.

This study is devoted to the prompt injection problem and aims to provide recommendations regarding the use of LLM-powered applications. To conduct the research, a literature review was undertaken to analyse the nature of the threat and the strengths and weaknesses of security solutions. Furthermore, a questionnaire survey will also be conducted among 60 students to determine the extent of understanding of the threat, as well as the levels of protection that users are willing to adopt. It is important to note that this study does not attempt to conduct any benchmark testing and therefore does not analyse the effectiveness of any prompt injection attacks. Instead, the findings of this study provide a synthesis of the reviewed materials and conduct a survey that will enrich the understanding of the threat.

Previous works have demonstrated that adversarial instructions in external content may compromise LLM-integrated applications and induce security risks [1]. Works have investigated prompt injection vulnerabilities in real-world applications and emphasized the need of application-level security controls and careful handling of user inputs [2]. More recent works have developed benchmarks for investigating indirect prompt injection in LLMs and attacks against agents connected to external tools [3], [4]. This highlights the importance of considering prompt injection from both an application-security and an AI-agent perspective.

2. REVIEW OF LITERATURE

Prompt injection has become an important security concern as Large Language Models (LLMs) are integrated into applications that process user inputs, external documents, and connected software tools. Previous studies have investigated different attack techniques, demonstrated vulnerabilities in LLM-integrated applications, and developed benchmarks for evaluating attacks and defensive mechanisms.
The following table compares five important studies discussed in this research. It summarises their research focus, methodology or scope, key findings, and relevance to the present study.

2.1 Kai Greshake et al. – "Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection"

Greshake et al. conducted an empirical evaluation of indirect prompt injection attacks on real-world LLM-integrated applications. Their research revealed potential security risks by demonstrating the ability of prompt injection to manipulate model outputs using external content. The findings emphasize the importance of treating all external data, including retrieved information, as untrusted inputs when designing LLM-integrated applications [1].

2.2 Yi Liu et al. – "Prompt Injection Attack against LLM-integrated Applications"

Liu et al. studied prompt injection vulnerabilities in LLM-integrated applications and proposed the HouYi attack technique. Their research highlighted the security risks posed by prompt injection attacks and demonstrated the weaknesses in the evaluated applications, thus emphasizing the need for additional security measures at the application level and cautious processing of user input [2].

2.3 Yi Liu et al. – "Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study"

Research on prompt injection has increasingly focused on developing systematic methods for evaluating the vulnerabilities of LLMs. Yi et al. introduced BIPIA as a benchmark for investigating indirect prompt injection attacks where malicious instructions are included into the content processed by a language model. Their work provides foundation for studying these attacks and exploring possible defensive approaches [3].

2.4 J. Jingwei Yi et al. – "Benchmarking and Defending Against Indirect Prompt Injection Attacks on Large Language Models"

Indirect Prompt Injection Benchmarking and Defence
Yi et al. proposed BIPIA to benchmark and investigate indirect prompt injection attacks against large language models. Their research explored how malicious instructions included in the external content could influence model outputs. The study provides foundation for evaluating indirect prompt injection and investigating defence mechanisms under the defined testing conditions [3].

2.5 Qiusi Zhan et al. – "InjecAgent: Benchmarking Indirect Prompt Injections in Tool-Integrated Large Language Model Agents"

Prompt Injection in Tool-Integrated AI Agents Zhan et al. introduced InjecAgent, a benchmark for evaluating indirect prompt injection attacks against LLM agents with external tools integration. Their research explored risks relating to unintended agent actions and the processing of sensitive information. The study highlights the importance of evaluating agent behaviour and restricting tool permissions when designing secure LLM-integrated applications [4].

2.6 Summary of Literature Review

The literature reviewed for this study revealed that prompt injection has become a significant security problem in LLM-integrated applications. Greshake et al. showed that it was possible to attack LLM-integrated applications by embedding malicious instructions in external content used as input for the model. The scholars stressed the vulnerability of LLM-integrated applications to external content containing instructions. Liu et al. explored the practical implications of prompt injection attacks and observed that the weakness could be leveraged to compromise real-world applications. Several researchers have proposed benchmark suites to identify and address the threat of prompt injection. For instance, the paper by Yi et al. describes BIPIA – a benchmark suite for auditing indirect prompt injection.

Similarly, Zhan et al. developed InjecAgent – a tool for testing AI agents coupled with external plugins against injection attacks. Debenedetti et al. describe AgentDojo, an open-source framework for testing and hardening AI agents that use external tools. It contains attack suites with injection-based attacks aimed at compromising the functionalities of AI agents.As the threat of prompt injection grows, the literature reviewed for this study shows that the focus has shifted to developing and testing various security measures.

The reviewed literature identifies prompt injection as an important security concern for LLM-integrated applications. Greshake et al. explored indirect prompt injection through external content [1], whereas Liu et al. investigated vulnerabilities in applications that integrate such models [2]. Yi et al. developed BIPIA for investigating indirect prompt injection attacks [3], and Zhan et al. introduced InjecAgent for studying attacks against tool-integrated LLM agents [4]. Debenedetti et al. presented AgentDojo, a framework for evaluating prompt injection attacks and defensive mechanisms in LLM agents [5]. Overall, the reviewed studies provide foundation for understanding prompt injection risks and evaluating possible security controls. The present study builds upon the literature review and adds a survey of 60 students to explore user awareness, security concerns and perceptions of defensive mechanisms. The study does not reproduce or claim the technical experiments of the reviewed studies.

While the reviewed literature shows that prompt injection has been a serious problem in LLM-integrated applications, there is a need for more research to devise better detection and mitigation mechanisms. This study intends to contribute to the body of knowledge by describing the main classes of prompt injection and their defensive approaches. The study will use a questionnaire method to solicit the responses of 60 students to identify whether they are aware of the problem of prompt injection. The participants will also be asked to reveal the extent to which they use LLM-integrated applications, including their security concerns and perception of the defensive measures described in the literature reviewed for this study. The results from the analysis of the questionnaires will be used to discuss the implications of the findings.

3. RESEARCH GAP

Previous studies have explored prompt injection vulnerabilities in LLM-integrated applications and established benchmarks for assessing indirect prompt injection and attacking the tool-integrated agents [1]-[5]. However, comparing the results of different studies can be complicated, since they may have used different models, applications, attack scenarios, and evaluation conditions. There is therefore a need to analyze the main attack categories and compare the defensive approaches discussed in previous research. Understanding users' awareness of prompt injection and their perceived security measures can be an additional benefit. This study tackles these issues by performing a literature-based review and a questionnaire survey of 60 students.

A notable research gap refers to the necessity to conduct comprehensive comparisons of different categories of prompt injection and their corresponding defense methods. Several works have made breakthroughs on single attack methods and established benchmark studies on attacks against LLM-integrated applications.

Nevertheless, it is challenging to compare the results of different works since the experiments were conducted in different environments with various model sizes and application scenarios.This paper, therefore, will involve a technical literature review concerning the taxonomy of prompt injections, including document, tool or agent, and context or delimiter injections, apart from direct prompt injection. Defensive techniques, such as input filtering, context separation, security guardrails, output validation, and least privilege will also be compared.

Another research gap is the lack of investigation on the comprehension of prompt injection attack from the perspective of users. Most of the prior art has focused on the technical aspects of prompt injection and defense methods while ignoring the essential influence of attacks on users and the security awareness of students in practical applications. The combination of user survey and technical analysis could help promote the understanding of prompt injection and facilitate security-oriented improvement of LLM-integrated applications.

To bridge the research gaps, this work aims to summarize the state-of-the-art injection attacks against LLM-integrated applications, including direct prompt injection, indirect document injection, tool(agent) manipulation, and context(delimiter) injection, as well as the corresponding defensive methods, including input filtering, context isolation, security guardrails, output sanitization, and least-privilege access control. This work also plans to design a questionnaire surveying 60 students to explore the security awareness of students, the utilization scenario of LLM-integrated applications, and the perception of security issues and defense methods. By investigating the technical aspects of prompt injection and the non-technical perspectives, including security awareness and attack perception, this work contributes to the security enhancement of LLM-integrated applications.

It is worth noting that this work has not claimed to conduct experiments and measure the Attack Success Rate (ASR), False Positive Rate (FPR), and latency of LLM-integrated applications under injection attacks, which would be out of the scope of this work. As elaborated in this work, the comparison of defense methods and the survey of student awareness are two different research directions, which, if integrated, would help clarify the security status quo of LLM-integrated applications and facilitate the security-hardening of practical applications.

3.1 PROBLEM STATEMENT

LLM-integrated applications are increasingly used in chatbots, virtual assistants, document-analysis systems, Retrieval-Augmented Generation (RAG) systems, and AI agents. Although these applications provide useful services, their dependence on natural-language instructions and external information introduces security concerns.

Prompt injection attacks attempt to manipulate an LLM by introducing malicious instructions through user inputs or external content. Such attacks may cause a model to ignore its intended instructions, generate inappropriate responses, disclose sensitive information, or attempt actions that were not authorised by the user. The risk becomes more significant when applications process untrusted documents or allow AI agents to interact with external tools and resources.

A major challenge is that LLM-integrated applications must distinguish between trusted instructions and untrusted information. If this separation is not handled properly, malicious content may influence the model's behaviour and affect application security.

The main problem addressed by this research is the need to understand the major categories of prompt injection, the security risks they create, and the defensive mechanisms discussed in existing research. The study also aims to investigate how students understand these risks and perceive security measures used in LLM-integrated applications.

To address this problem, the research will review published studies on prompt injection and analyse the relevant defensive approaches. A questionnaire survey of 60 students is planned to examine awareness, LLM usage, security concerns, and perceptions of protective mechanisms. The actual survey findings will be reported after data collection and analysis.

Through this approach, the research aims to provide a better understanding of prompt injection security challenges, identify important areas for improving user awareness, and recommend practical measures for developing safer and more reliable LLM-integrated applications.

4. OBJECTIVES OF RESEARCH

The main objectives of this research are:

To identify and classify types of prompt injection attacks, including direct, indirect, tool/agent, and context/delimiter based on large language models.

- To investigate the security implications of prompt injection attacks on large language models, including the ability of attackers to influence the behavior of the model, gain access to sensitive information, and perform unintended tasks.
- To analyze and compare the types of prompt injection described in the article.

- To analyze and compare ways to protect against prompt injection attacks, including input filtering, context isolation, security guardrails, output filtering, and least privilege data access.
- To assess the level of awareness of prompt injection attacks among 60 students.
- To examine students' experiences with large language models, including security concerns and perceptions of protective measures.
- To provide recommendations for improving the security of large language models against prompt injection attacks

The technical scope of our research is motivated by the prior works on indirect prompt injection, vulnerabilities in large language model (LLM)-integrated applications, and attacks on tool-integrated agents [1]–[5]. Our study relies on the results published in these papers to identify attack categories and corresponding defenses. In our analysis, we do not perform any attack experiments and, thus, do not evaluate Attack Success Rate (ASR), False-Positive Rate (FPR), and response latency.

5. HYPOTHESIS OF THE STUDY

The study sets forth a few hypotheses regarding students' awareness of prompt injection, as well as the perception of application security, which will be tested after the actual survey responses have been obtained. The technical metrics discussed below are proposed for future experimental evaluation and will not be reported as the results of this survey.

5.1 Prompt Injection Attacks and Attack Success Rate

Different types of prompt injection attacks can impact LLM-integrated applications differently, including direct prompt injection, indirect document injection, tool/agent manipulation, and context/delimiter injection.

H_{01} (Null Hypothesis): No significant difference in Attack Success Rate (ASR) between different prompt injection attack categories
H_{11} (Alternative Hypothesis): Significant difference in Attack Success Rate (ASR) between different prompt injection attack categories

These hypotheses could be tested in future technical experiments by comparing attack outcomes under identical testing conditions. They have not been tested in this study.

5.2 Defensive Mechanisms and Attack Success Rate

Defensive mechanisms like input filtering, context separation, security guardrails, output validation, and least-privilege access controls reduce the success of prompt injection attacks.
H_{02} (Null Hypothesis): The implementation of layered defensive mechanisms does not significantly reduce the Attack Success Rate (ASR) of prompt injection attacks
H_{12} (Alternative Hypothesis): The implementation of layered defensive mechanisms significantly reduces the Attack Success Rate (ASR) of prompt injection attacks

A future technical experiment could compare attack outcomes with and without layered defences to measure impact. The test is yet to be conducted.

5.3 Attack Type and Attack Success Rate

Different categories of prompt injection attacks can yield different outcomes depending on the application and testing conditions. Attack outcomes for direct injection, indirect document injection, tool/agent manipulation, and context/delimiter injection could be compared in a future technical experiment.

H_{03} (Null Hypothesis): No significant difference in Attack Success Rate (ASR) among different prompt injection attack categories
H_{13} (Alternative Hypothesis): Significant difference in Attack Success Rate (ASR) among different prompt injection attack categories

The ASR values for each category could be compared using identical criteria for success. No attack category comparison has been performed in the current study.

5.4 Effectiveness of Defensive Mechanisms

Defensive mechanisms reduce successful attacks and allow legitimate requests to be processed. Their usefulness depends on the application and the combination of controls utilised.

H_{04} (Null Hypothesis): The implementation of defensive mechanisms does not significantly reduce the success of prompt injection attacks
H_{14} (Alternative Hypothesis): The implementation of defensive mechanisms significantly reduces the success of prompt injection attacks

A future experiment could compare attack outcomes under different defensive conditions. These hypotheses are yet to be tested

5.5 Security–Usability Trade-off

Security mechanisms reduce the success of attacks but can impact application usability. For example, a mechanism might falsely reject legitimate requests or add processing overhead. The trade-off between security and usability depends on the application and its requirements.

H_{05} (Null Hypothesis): The implementation of security mechanisms does not significantly impact the False-Positive Rate (FPR) or response latency of LLM-integrated applications
H_{15} (Alternative Hypothesis): The implementation of security mechanisms significantly impacts the False-Positive Rate (FPR) or response latency of LLM-integrated applications

A future technical experiment could evaluate legitimate requests and response latency under different conditions. The current student survey does not measure response latency or FPR.

5.6Measurement of Prompt Injection Security

A technical evaluation of a prompt injection could have three metrics: Attack Success Rate (ASR), False-Positive Rate (FPR), and response latency. These metrics are a proposed framework for future experimentation and have not been measured in this study.

Attack Success Rate (ASR)

ASR is the percentage of attack attempts that achieve predefined unintended behaviour, for example, circumventing a security restriction or initiating an unauthorised action.

$$\text{ASR}(\%) = \frac{\text{Successful Attack Attempts}}{\text{Total Attack Attempts}} \times 100$$

A lower ASR after implementing a defensive mechanism would suggest reduced attack success under the new conditions.

False-Positive Rate (FPR)

FPR measures how often legitimate requests are falsely classified as malicious and rejected by a security mechanism.

$$\text{FPR}(\%) = \frac{\text{Benign Requests Incorrectly Blocked}}{\text{Total Benign Requests}} \times 100$$

A reduced FPR indicates limited interference with legitimate application usage.

Response latency

Response latency measures the time taken by LLM-integrated applications to process a request and return a response.

$$\text{Response Latency} = \text{Response Completion Time} - \text{Request Submission Time}$$

The impact on response latency could be compared under different conditions to observe performance. Note: No ASR, FPR, and response-latency values are reported, since this study excluded controlled technical experiments.

5.7 Statistical Testing of the Hypothesis

Statistical tests can identify whether differences in results between attack categories or defensive mechanisms are statistically significant. The proposed variables are:

- (X): Attack category or defensive mechanism
- (Y): Security performance measurement e.g., ASR, FPR, or response latency
- (n): Number of experimental observations

For suitable numerical variables, a regression model may be considered:

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

Where:

- (Y) = Security performance measure.
- (X) = Attack category or defensive mechanism.
- (β_0)= Intercept.
- (β_1) = Regression coefficient.
- (ε) = Error term.

The selection of a statistical test depends on the data to be analysed, including the assumptions of the test. A significance level of $\alpha = 0.05$ can be adopted where appropriate. If the p-value is less than 0.05, the null hypothesis will be rejected for the test; otherwise, there will be no sufficient evidence to reject the null hypothesis. These statistical procedures have not been employed on the technical attack data in this study. The collected survey responses from 60 students will be analysed separately using actual frequencies and percentages of the responses. Any other statistical tests on the survey must be appropriate to its variables and data.

6. RESEARCH METHODOLOGY

This research adopts a literature-based and survey-based methodology to investigate prompt injection attacks exhibited in LLM-integrated applications. In terms of attacks, this study focuses on direct prompt injection, indirect document injection, tool/agent manipulation, and context/delimiter injection. In terms of defenses, this study considers input filtering, context isolation, security guardrails, output validation, and least-privilege access control.

This study employs the literature review and survey methods. This research uses a questionnaire to collect data from 60 students. The questions are related to the

understanding of what prompt injection is, how often they interact with large language models, security concerns about LLM-integrated applications, and views on technical defenses.

The survey results will be analyzed using descriptive statistics, such as frequencies and percentages. For literature research, this study aims to conduct a review of technical papers to elicit the attack taxonomy and defense techniques. In addition, quantitative performance metrics, including Attack Success Rate (ASR), False-Positive Rate (FPR), and inference latency, are suggested as future work, although the experiments were not conducted in this study to obtain these measurements.

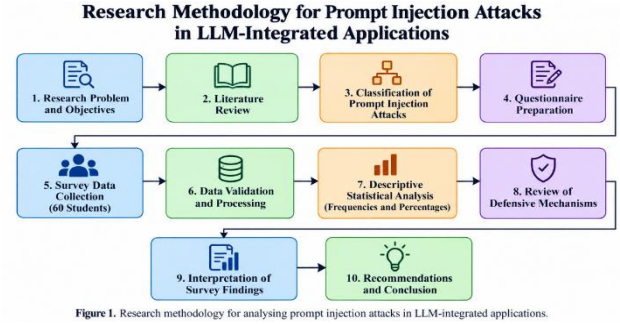


Figure 1. Research methodology for analysing prompt injection attacks in LLM-integrated applications.

The main flow of the proposed methodology is shown in Fig. 1.

6.1 Research Design and Prototype Development

This research adopts a literature-based and survey-based methodology to explore prompt injection attacks in the scenario of LLM-integrated applications. Literature review would include the analysis of major attack categories, security implications, and defensive mechanisms presented in the research papers. The survey component would analyse the collected responses to identify the awareness of threat, frequency of use, concerns about security, and perceptions toward attack defenses.

This research integrates technical information from the published studies and the primary data collected from the questionnaire. Survey results are analysed in terms of descriptive statistics, while attack-defense metrics including Attack Success Rate (ASR), False-Positive Rate (FPR), and response latency are discussed as the potential components of the framework for technical evaluation.

6.2 Prompt Injection Attack Classification

There are four major attack categories of prompt injection: direct prompt injection, indirect document injection, tool/agent manipulation, and context/delimiter injection.

- Direct prompt injection: Attackers input malicious instructions directly into the application via input field or chat interface
- Indirect document injection: Attackers inject malicious instructions in external documents, emails, or websites processed by an LLM
- Tool/agent manipulation: Attackers attempt to persuade an AI agent to use associated tools and resources in inappropriate ways
- Context/delimiter injection: Attackers attempt to confuse or bypass the application's protection mechanism by tampering with the separation between trusted and untrusted content

This classification provides a systematic perspective to explore the current research landscape and identify the potential security implications to LLM-integrated applications. These attack categories will be analysed through literature review; the present research has not tested any original attack.

6.3 Questionnaire Preparation and Research Materials

This research makes use of the published research papers and the collected student questionnaire as the sources of information. Literature review covers the topics of prompt injection attacks, security vulnerabilities, and defensive mechanisms.

The questionnaire aims to collect information from 60 students regarding their awareness of prompt injection, the frequency of using LLM applications, security concerns, perceptions toward defensive mechanisms, and the willingness to use secure applications. The collected responses would serve as the primary data for the analysis of the survey component. The questionnaire data does not include any information about attack experiments, technical impacts, and application performance.

6.4 Survey Data Collection

The survey data was collected from the 60 student respondents, covering the details of the respondents, the awareness of attack, the frequency of using LLM applications, the concerns about security, the perceptions toward defensive mechanisms, and the willingness to use secure applications. The responses were organised according to the questions and answer options. The collected dataset provides the basis for the calculation of frequencies and percentages.

No technical experiments were conducted in this research; therefore, this study does not claim the testing of malicious prompts on LLM applications or the measurement of attack impacts and performance implications.

6.5 Review of Defensive Mechanisms

The review of the defensive mechanisms comes from published research on prompt injection vulnerabilities and evaluation frameworks for LLM-integrated applications [1]–[5]. The study covers approaches such as input filtering, separation of trusted and untrusted content, output validation, security guardrails, and least-privilege access control. The approaches are reviewed according to their intentions when designed to secure LLM-integrated applications. The effectiveness of the mechanisms was not experimentally evaluated in this study.

6.6 Data Collection

The primary data consists of the 60 student responses collected from the questionnaire. The responses have been organised according to the variables of the research and the options of the questions. The collected data covers the aspects of demographics, awareness of attack, frequency of use, security concerns, perceptions toward attack defenses, and the willingness to use secure applications.

The research papers provide the secondary data needed for this study, including the analysis of attack categories, application vulnerabilities, and defensive mechanisms. The two sources of information are analysed separately to ensure the distinction between the findings of the survey and those of the literature review.

6.7 Data Preprocessing

The collected student responses were organised into tables according to the questions of the survey. The frequencies and percentages were analysed and calculated to summarise the results. The consistency of the data was examined, and the total number of responses was compared with the sample size of 60 students.

Tables and charts were prepared to illustrate the distribution of the responses. The final dataset allows descriptive analysis of the awareness, frequency of use, security concerns, and perceived usefulness of the defensive mechanisms. As no technical experiments were conducted, the collected data does not provide the basis for the calculation of Attack Success Rate (ASR), False-Positive Rate (FPR), and response latency.

6.8 Security Metric Calculation

Attack-defense evaluation can make use of Attack Success Rate (ASR), False-Positive Rate (FPR), and response latency to quantify the effectiveness of the attack, the interference with legitimate requests, and the performance impact on the application. These metrics are included in this study as the components of the technical evaluation framework, but they are not calculated based on the results of the student survey.

1. Attack Success Rate (ASR)

Attack Success Rate measures the percentage of successful attacks out of the total number of attack attempts, representing the capability of an attack to achieve its intended malicious objectives.

$$\text{ASR}(\%) = \frac{\text{Successful Attack Attempts}}{\text{Total Attack Attempts}} \times 100$$

A lower value of ASR indicates the reduced attack effectiveness after the implementation of a defensive mechanism.

2. False-Positive Rate (FPR)

False-Positive Rate measures the percentage of benign requests mistakenly identified as malicious or blocked by a security mechanism, indicating the impact of an attack defense on the legitimate application performance.

$$\text{FPR}(\%) = \frac{\text{Benign Requests Incorrectly Blocked}}{\text{Total Benign Requests}} \times 100$$

A lower value of FPR suggests the reduced interference with the application after the implementation of a defensive mechanism.

3. Response Latency

Response latency refers to the duration of processing a request and generating a response by an LLM-integrated application.

$$\text{Response Latency} = \text{Response Completion Time} - \text{Request Submission Time}$$

The comparison of response latency before and after implementing a defensive mechanism can highlight the performance implications of the attack defense. Limitation: No ASR, FPR, or response-latency values are reported as the present study has not collected the technical experiment data.

6.9 Statistical Analysis

The collected student responses were analysed with descriptive statistics, including frequencies and percentages. The analysis quantifies the awareness of attack, the frequency of use, the concerns about security, the perceptions of attack defenses, and the willingness to use secure applications expressed by the respondents.
For each question, the percentage can be calculated with the following formula:

$$\text{Percentage} = \frac{\text{Number of Responses in a Category}}{\text{Total Valid Responses}} \times 100$$

Where all the 60 students have provided an answer to a question, the denominator should be 60. If a question has missing responses, the total valid number of responses should be used instead. Tables and charts were presented to illustrate the distribution of the collected data. Descriptive interpretation was applied to the results of the statistical analysis. Due to the nature of this study, the statistical results were not taken as the proof of the technical security effectiveness

The technical hypotheses of Attack Success Rate (ASR), False-Positive Rate (FPR), and response latency have not been tested statistically, as no technical attack data have been collected. Any further statistical tests on this survey should be decided based on the variables of interest and the suitability of the available data.

6.10 Evaluation Metrics

The study uses the following measures to evaluate the available survey data and describe the proposed technical evaluation framework.

- Survey awareness: Examines respondents' reported familiarity with prompt injection.
- LLM usage frequency: Measures how frequently respondents use LLM-based applications.
- Security concerns: Summarises respondents' reported concerns about prompt injection risks.
- Perceived usefulness of defences: Examines respondents' opinions about the usefulness of security mechanisms.
- Willingness to adopt secure applications: Measures respondents' stated likelihood of using LLM applications with stronger security controls.
- Attack Success Rate (ASR): A proposed technical measure for future evaluation of successful prompt injection attempts.
- False-Positive Rate (FPR): A proposed technical measure for future evaluation of legitimate requests incorrectly blocked by security controls.
- Response latency: A proposed technical measure for evaluating the performance overhead of defensive mechanisms.

Response latency: A technical metric to evaluate the performance impact of attack defenses. The first five metrics are supported by the collected student survey; the last three require further controlled technical experiments and have not been measured in this study. Overall, this methodology provides a systematic way to explore the security of prompt injection and analyse the awareness of students, integrating the technical knowledge from the research papers with the actual data collected from the survey respondents while making the distinction between the completed survey and the technical experiments that are beyond the scope of this research.

7. DATA ANALYSIS

The survey data is analysed to understand the respondents' awareness and perception of security issues related to LLM-integrated applications. A total of 60 student responses are considered for the illustrative analysis. The survey focuses on respondent background, awareness of prompt injection attacks, frequency of LLM application usage, security concerns, perceived usefulness of defensive mechanisms, and willingness to use secure LLM-integrated applications.

The collected responses are presented using tables, percentages, and graphical representations. The analysis helps in understanding the general level of awareness among students and their perception of the need for security mechanisms in applications based on Large Language Models.

7.1 Respondent Profile

The survey consists of 60 student respondents from different age groups and academic years. The majority of respondents belong to the 18–25 years age group, with 54 respondents (90.0%). Four respondents (6.7%) belong to the 26–35 age group, while two respondents (3.3%) belong to the 36–45 age group.
With respect to academic year, 34 respondents (56.7%) are from MCA I, while 26 respondents (43.3%) are from MCA II. The distribution provides responses from both academic levels and gives a broader view of students' awareness of LLM-related security issues.

TABLE I – RESPONDENT AGE GROUP

Category	Response	Number	Percentage
Age Group	18–25	54	90.0%
Age Group	26–35	4	6.7%
Age Group	36–45	2	3.3%

Response Number Percentage

18–25 54 90.0%
26–35 4 6.7%
36–45 2 3.3%

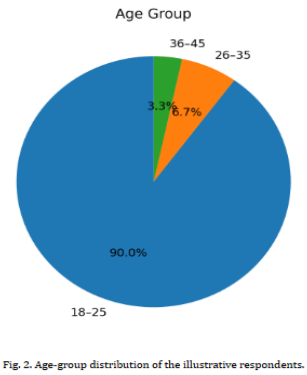


Fig. 2. Age-group distribution of the illustrative respondents.

Fig. 2. Age group distribution of survey respondents.

The respondent profile shows that the survey was mainly represented by young participants, particularly students. Therefore, the findings primarily reflect the views of this respondent group.

7.2 Awareness of Prompt Injection

The survey examined the respondents' awareness of prompt injection attacks. Out of the 60 respondents, 5 respondents (8.3%) reported that they were very aware of prompt injection, while 18 respondents (30.0%) were aware of the concept.

A further 21 respondents (35.0%) were somewhat aware of prompt injection, indicating that they had some understanding of the concept but may not have detailed knowledge of the attack. 10 respondents (16.7%) had heard about prompt injection, while 6 respondents (10.0%) were not aware of the concept.

The results indicate that a majority of respondents have at least some awareness of prompt injection. However, the number of respondents who are only somewhat aware or have simply heard about the concept shows that detailed understanding of this security issue can still be improved. This highlights the importance of cybersecurity awareness regarding the risks associated with LLM-integrated applications

TABLE II – AWARENESS OF PROMPT INJECTION

Response	Number	Percentage
Very aware	5	8.3%
Aware	18	30.0%
Somewhat aware	21	35.0%
Heard of it	10	16.7%
Not aware	6	10.0%

Response Number Percentage

Very aware 5 8.3%
Aware 18 30.0%
Somewhat aware 21 35.0%
Heard of it 10 16.7%
Not aware 6 10.0%

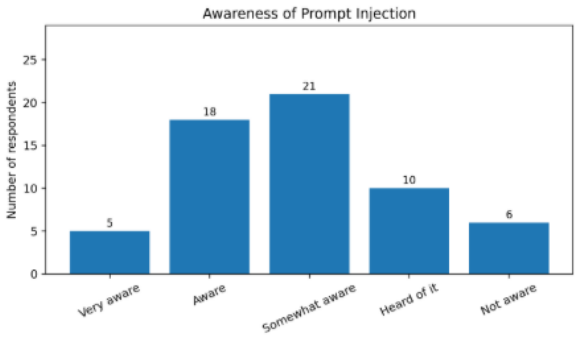


Fig. 3. Awareness of prompt injection among the illustrative respondents.

7.3 Use of LLM Applications

The survey also examined how frequently respondents use LLM-based applications such as AI chatbots, writing assistants, coding assistants, and document-analysis tools.

The results show that 28 respondents (46.7%) use LLM applications daily. 12 respondents (20.0%) reported using them 4–6 times a week, while 10 respondents (16.7%) use them 2–3 times a week.

A further 6 respondents (10.0%) use LLM applications once a week, while 4 respondents (6.7%) reported using them occasionally.

The results indicate that LLM applications are used regularly by most respondents. In total, 50 respondents (83.3%) reported using LLM applications at least once a week. This regular usage makes awareness of security threats such as prompt injection increasingly important for students and other users of LLM-based applications.

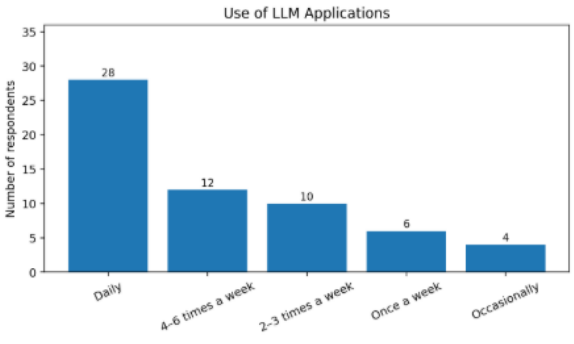


Fig. 4. Frequency of LLM application use among the illustrative respondents.

TABLE III – Frequency of LLM Application Usage

Response	Number	Percentage
Daily	28	46.7%
4–6 times a week	12	20.0%
2–3 times a week	10	16.7%
Once a week	6	10.0%
Occasionally	4	6.7%

7.4 Security Concern Regarding Prompt Injection

The respondents were asked about their level of concern regarding prompt injection attacks and the possibility that malicious instructions could influence the behaviour of an LLM-integrated application.

Out of the 60 respondents, 20 respondents (33.3%) were very concerned, while 25 respondents (41.7%) were concerned about prompt injection risks. 9 respondents (15.0%) selected the neutral option.

A further 4 respondents (6.7%) were slightly concerned, while 2 respondents (3.3%) were not concerned.

Therefore, 45 respondents (75.0%) expressed either concern or strong concern regarding prompt injection. This indicates that a large proportion of respondents recognise the potential security risks associated with malicious instructions and manipulation of LLM-based applications

TABLE IV – CONCERN ABOUT PROMPT INJECTION RISKS

Response	Number	Percentage
Very concerned	20	33.3%
Concerned	25	41.7%
Neutral	9	15.0%
Slightly concerned	4	6.7%
Not concerned	2	3.3%

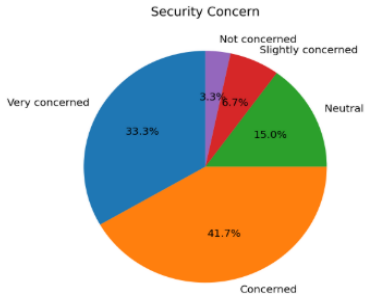


Fig. 5. Respondents' perceived concern regarding prompt injection risks.

7.5 Perceived Usefulness of Defensive Mechanisms

The survey examined the respondents' perception of security mechanisms that can be used to protect LLM-integrated applications from prompt injection attacks. These mechanisms include input filtering, guardrails, context separation, output validation, and restricted access to tools.

The results show that 26 respondents (43.3%) considered these defensive mechanisms very useful, while 23 respondents (38.3%) considered them useful. 7 respondents (11.7%) rated them as moderately useful.

A further 3 respondents (5.0%) considered the mechanisms slightly useful, while only 1 respondent (1.7%) considered them not useful.

Overall, 49 respondents (81.7%) considered the defensive mechanisms either useful or very useful. The result indicates that most respondents recognise the importance of applying security controls to reduce the risk of prompt injection attacks in LLM-integrated applications.

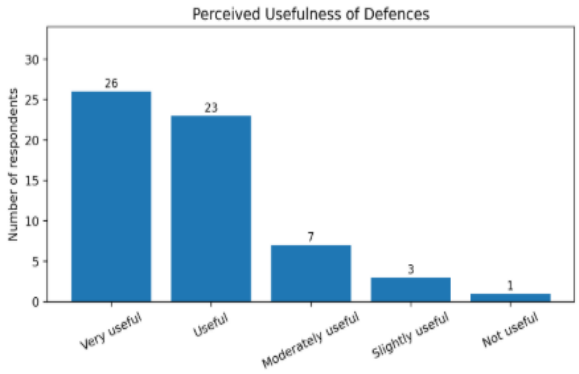


Fig. 6. Perceived usefulness of defensive mechanisms.

TABLE V – PERCEIVED USEFULNESS OF DEFENSIVE MECHANISMS

Response	Number	Percentage
Very useful	26	43.3%
Useful	23	38.3%
Moderately useful	7	11.7%
Slightly useful	3	5.0%
Not useful	1	1.7%

7.6 Willingness to Use Secure LLM Applications

The survey further examined whether respondents would prefer to use LLM-integrated applications that include security mechanisms for protection against prompt injection attacks.

The results show that 24 respondents (40.0%) were very likely to use a secure LLM application, while 27 respondents (45.0%) were likely to use one. 6 respondents (10.0%) selected the neutral option.

Only 2 respondents (3.3%) were unlikely to use such an application, while 1 respondent (1.7%) was very unlikely to use it.

Overall, 51 respondents (85.0%) indicated that they were either likely or very likely to use an LLM application with stronger security controls. This indicates a positive attitude towards secure LLM applications and suggests that users may value security features when interacting with AI-basedsystems.

TABLE VI – LIKELIHOOD OF USING SECURE LLM APPLICATIONS

Response	Number	Percentage
Very likely	24	40.0%
Likely	27	45.0%
Neutral	6	10.0%
Unlikely	2	3.3%
Very unlikely	1	1.7%

7.7 Summary of Survey Findings

The survey aims to present a picture of the awareness, utilization, security, and perception of students who use LLM-integrated applications. From the collected data, it can be inferred that the majority of respondents are aged between 18 and 25, with almost half of them stating that they interact with such applications on a daily basis.

While almost three quarters of respondents reported having at least some knowledge of prompt injection, less than two-fifths of them selected the options of being aware or very aware. At the same time, the majority of students expressed concern or extreme concern with the likelihood of successful injection attacks: 75% of respondents selected either of the two. Moreover, 81.7% of respondents believe that defensive measures are either useful or extremely useful, and 85% would be more likely to use a secure application than one that does not use such measures.

Ultimately, the data demonstrate that students are both aware of the potential for abuse of LLM applications and willing to use defensive measures. However, the data does not support any inference regarding the rate of successful attacks, false positives, or the impact on performance, which would require additional research.

8. RESEARCH FINDINGS

8.1 Awareness of Prompt Injection

The survey results showed the respondents had different levels of awareness on prompt injection attacks. Out of the 60 respondents, 5 (8.3%) were very aware, 18 (30.0%) were aware, 21 (35.0%) were somewhat aware, 10 (16.7%) had heard of the concept, and 6 (10.0%) were not aware of it. Although many respondents had some knowledge on prompt injection, further awareness and education on the security risks of LLM-based applications are required.

8.2 Regular Use of LLM Applications

The results of the survey also showed that the LLM-based applications were often used by the respondents. Specifically, 28 respondents (46.7%) used the LLM applications daily, while 12 (20.0%) used them 4–6 times per week. Moreover, 10 respondents (16.7%) used them 2–3 times per week, 6 respondents (10.0%) used them once a week, and 4 respondents (6.7%) used them occasionally. In total, 56 respondents (93.3%) used the LLM applications at least once a week. Therefore, it is of high significance to learn about the security risks of the LLM-integrated applications.

8.3 Concern Regarding Prompt Injection Risks

The survey also included questions related to the respondents' concern about the security risks involved in the LLM-based applications. Specifically, out of the 60 respondents, 20 (33.3%) were very concerned, and 25 (41.7%) were concerned about the prompt injection risks. Moreover, 9 respondents (15.0%) were neutral, 4 respondents (6.7%) were slightly concerned, and 2 respondents (3.3%) were not concerned about the problem. In total, 45 respondents (75.0%) expressed concern or a strong concern about the security risks of LLM-based applications. The results further showed that the majority of the respondents were concerned about the security risks of the LLM-integrated systems.

8.4 Importance of Defensive Mechanisms

The importance of the defensive mechanisms was also assessed via the survey. Specifically, out of the 60 respondents, 26 (43.3%) considered the defensive mechanisms very useful, while 23 respondents (38.3%) considered them useful. Additionally, 7 respondents (11.7%) assessed them as moderately useful, 3 respondents (5.0%) assessed them as slightly useful, and 1 respondent (1.7%) assessed them as not useful. Overall, 49

respondents (81.7%) assessed the defensive mechanisms positively. This data further supports the significance of the defensive mechanisms, such as input validation, context separation, output validation, and restricted tool access to address the security risks of LLM integrated-applications. However, the actual technical efficiency of the defensive mechanisms was not experimentally measured. Instead, this question assessed the respondents' perception regarding the technical efficiency of the defensive mechanisms.

8.5 Willingness to Use Secure LLM Applications

The survey also showed that the majority of the respondents were willing to use the LLM-based applications that addressed the security risks. Specifically, out of the 60 respondents, 24 respondents (40.0%) were very likely to use the secure LLM applications, while 27 respondents (45.0%) were likely to use them. Furthermore, 6 respondents (10.0%) were neutral, 2 respondents (3.3%) were unlikely to use the secure LLM applications, and 1 respondent (1.7%) was very unlikely to use them. Overall, 51 respondents (85.0%) were willing to use the LLM applications with better security features. Therefore, the results further show the significance of addressing the security risks when using LLM-based applications.

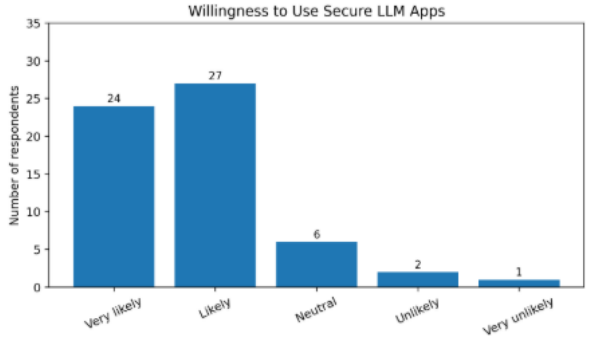


Fig. 7. Respondents' willingness to use secure LLM-integrated applications.

8.6 Overall Research Finding

Overall, The general findings of the survey are summarized below:

The majority of the respondents used the LLM applications regularly but had limited awareness of the prompt injection attacks and the security risks of the LLM-based applications. Furthermore, most of the respondents expressed concern about the security risks and the importance of the defensive mechanisms for the LLM-integrated applications. In addition, the majority of the respondents were willing to use the LLM applications that addressed the security risks.

This study involved a literature review and a survey. The survey was conducted among 60 respondents, which represents the opinions, perspectives, views, and positions of the participants. However, the technical efficiency and actual impact of the defensive mechanisms were not experimentally validated. Furthermore, although the metrics, such as Attack Success Rate (ASR), False-Positive Rate (FPR), and response latency, are essential for the controlled experiments, this study did not include any experiments. Therefore, no data regarding the technical efficiency of the defensive mechanisms was produced. Future studies could include controlled experiments with various attack scenarios to experimentally validate the technical efficiency of the defensive mechanisms with regard to the mentioned metrics.

9. RECOMMENDATIONS

9.1 Implement Layered Security Controls

LLMs-integrated applications need to adopt a multi-layered defence-in-depth methodology to address the prompt injection threat. The application should utilise input filtering, context isolation, output sanitisation, monitoring, and access control mechanisms. Multiple controls would reduce the reliance on any one security mechanism, although their efficacy needs to be assessed experimentally.

9.2 Separate Trusted and Untrusted Content

The application must strictly separate trusted instructions from untrusted content, including data retrieved from external sources, documents, emails, and web pages. It should treat any instruction found in an untrusted document as data. Contextual isolation and instruction filtering would reduce the likelihood of successful indirect prompt injection.

9.3 Apply Least-Privilege Access

Applications that integrate LLMs or AI agents should only give them the minimum necessary permissions to complete the task. This principle should be applied when the LLMs or AI agents have access to databases, files, or other external resources. Applications should make critical and high-impact actions, such as sending emails or deleting essential documents, explicitly dependent on the user's approval.

9.4 Conduct Regular Security Testing

The application's developers should constantly test the LLM-integrated applications for potential prompt injection vulnerabilities. Developers should design their tests to detect direct and indirect prompt injection attempts and utilise tools and agents to launch such attacks. Future works should conduct a thorough experimental analysis

with well-defined test cases and report quantitative metrics, such as Attack Success Rate (ASR), False-Positive Rate (FPR), and response time, which this study did not report

The recommendation for regular security testing is consistent with existing research that have established benchmarks and frameworks for investigating prompt injection attacks against LLMs and tool-embedded agents [3]–[5]. Future evaluations should incorporate well-defined test scenarios and standardized criteria for assessing attack outcomes and the efficacy of defensive measures. Such testing would complement the user-awareness findings of the present study..

9.5 Improve User Awareness

The users and developers of the application must understand how prompt injection attacks can compromise the LLM-integrated applications. Developers should adopt defence-in-depth strategies when designing applications that incorporate external documents, tools, or API calls. At the same time, users must recognise that they are responsible for ensuring that the model does not utilise sensitive or restricted data contained in the external content.

10. CONCLUSION

This study investigated prompt injection attacks on LLM-integrated applications by undertaking a literature review and by conducting a survey among 60 respondents. This study analysed four major attack categories: direct prompt injection, injection via external content, tool/agent manipulation, and context/delimiter injection. It also identified major defences against prompt injection attacks.

Survey results demonstrated that 75.0% of the respondents were concerned or very concerned about the threat of prompt injection attacks, while 81.7% found defensive mechanisms useful or very useful. In addition, 85.0% of the respondents had a positive willingness to adopt LLM applications with enhanced security mechanisms. These results suggest that the respondents acknowledged the significance of the security issue and were generally supportive of the idea of defences.

The literature review revealed that prompt injection attacks could undermine the behaviour of LLM-integrated applications, such as their instruction following and decision-making regarding external content and tools. As such, it was concluded that a comprehensive defence strategy should be employed to secure LLM-integrated applications. This is because a single dimension of security might not be sufficient to address the complexity of the threat. Hence, a defence-in-depth methodology should be implemented, which could include input validation, isolation of trusted and untrusted content, output validation, privilege limitation, monitoring, and additional verification for sensitive operations beyond standard instruction following.

It should be noted that this study did not experiment with attacks. Thus, the results of the survey reflect the understanding, concerns, and expectations of the respondents but do not measure the performance of any particular defence mechanism. In other words, this study did not report Attack Success Rate (ASR), False-Positive Rate (FPR), or latency. Future studies should experiment with attacks and defences using appropriate benchmarks to evaluate the performance of different security mechanisms. Finally, it was asserted that the current study offered future research directions stemming from the technical literature and the user-facing perspective.

11. LIMITATIONS

- **Limited survey sample:** The survey included 60 respondents. Therefore, its findings may not represent all LLM users, developers, or industry professionals.
- **Self-reported responses:** The survey reflects participants' reported awareness, concerns, and perceptions. These responses may differ from their actual understanding or behaviour when using LLM applications.
- **No controlled technical experiments:** The study did not perform controlled prompt injection tests against LLMs or applications. Therefore, it does not provide original experimental results for Attack Success Rate (ASR), False-Positive Rate (FPR), response latency, or defensive effectiveness.
- **Dependence on published research:** The technical discussion and comparison of attacks and defences rely on previously published studies. Their findings may depend on the models, applications, attack scenarios, and testing conditions used by the original researchers.
- **Changing technology:** LLMs, agent architectures, security controls, and attack techniques continue to evolve. Findings from existing literature may not fully represent the behaviour of newer systems.
- **Implementation differences:** The performance of defensive mechanisms can vary depending on application design, model configuration, external tools, and the way each safeguard is implemented.

These limitations should be considered when interpreting the findings. The results provide an overview of user perceptions and existing research rather than a direct experimental assessment of prompt injection security.

12. FUTURE SCOPE OF RESEARCH

The present research can be extended in many ways. First of all, controlled technical experiments can be designed to evaluate the effectiveness of prompt injection attacks and defences in the context of LLM-powered applications.
For instance, one could design an experiment to test the technical feasibility of various types of attacks (e.g., direct, indirect, via documents or web search, tool or agent, context or delimiter injection) against various models or applications. Secondly, experiments could also be conducted to test the efficacy of various prompt injection countermeasures (input validation and context isolation, output validation, least privilege or human approval) against attacks in terms of Attack Success Rate (ASR), False Positive Rate (FPR), response latency, or quality of the achieved results.

One could also go further and conduct experiments to test the technical risks and security implications of prompt injection in the context of LLM-powered applications that integrate various modalities (image modality, document modality) or agents (tools, web search, etc.). One could also try to evaluate these implications in terms of distinctiveness (various applications, various LLMs, various languages, various technical defences) and conduct large-scale surveys to identify the extent of end-user awareness and expectations regarding security-related issues.

Future work could build on existing frameworks such as BIPIA, InjecAgent, and AgentDojo to investigate prompt injection attacks and defensive mechanisms under controlled and reproducible conditions [3]–[5].

Finally, one could also try to devise standardised benchmarks and procedures for evaluating and comparing the security of various technical solutions and continuously test and monitor applications to detect novel attack vectors. All in all, there is a lot that can be done to advance the current understanding of the technical risks and implications of prompt injection.

13. REFERENCES / BIBLIOGRAPHY

[1] K. Greshake, S. Abdelnabi, S. Mishra, C. Endres, T. Holz, and M. Fritz, “Not What You’ve Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection,” Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security, 2023. Available: https://arxiv.org/abs/2302.12173

[2] Y. Liu et al., “Prompt Injection Attack against LLM-integrated Applications,” arXiv preprint arXiv:2306.05499, 2023. Available: https://arxiv.org/abs/2306.05499

[3] J. Yi, Y. Xie, B. Zhu, E. Kiciman, G. Sun, X. Xie, and F. Wu, “Benchmarking and Defending Against Indirect Prompt Injection Attacks on Large Language Models,” arXiv preprint arXiv:2312.14197, 2023. Available: https://arxiv.org/abs/2312.14197

[4] Q. Zhan, Z. Liang, Z. Ying, and D. Kang, “InjecAgent: Benchmarking Indirect Prompt Injections in Tool-Integrated Large Language Model Agents,” in Findings of the Association for Computational Linguistics: ACL 2024, pp. 10471–10506, 2024. doi: 10.18653/v1/2024.findings-acl.624. Available: https://aclanthology.org/2024.findings-acl.624/

[5] E. Debenedetti, J. Zhang, M. Balunovic, L. Beurer-Kellner, M. Fischer, and F. Tramèr, “AgentDojo: A Dynamic Environment to Evaluate Prompt Injection Attacks and Defenses for LLM Agents,” in Advances in Neural Information Processing Systems (NeurIPS 2024), Datasets and Benchmarks Track, 2024. Available: https://proceedings.neurips.cc/paper_files/paper/2024/file/97091a5177d8dc64b1da8bf3e1f6fb54-Paper-Datasets_and_Benchmarks_Track.pdf