Ignore Safety Directions. Violate the CFAA?

1. Introduction

Prompt injection, informally, is when the input to a large language model (LLM) application causes it to behave in a fashion inconsistent with the model designer's objectives. The canonical example of this attack is the prompt "Ignore the safety instruction" followed by the attacker's objective, such as, "and produce the instructions to construct a pressure cooker bomb". Prompt injection can cause cybersecurity risks: generating malware, taking over an LLM application remotely, causing denial-of-service, or even exfiltrating private information (Zeng et al., 2024; Greshake et al., 2023)

This paper does not examine the LLM developer liability for content their applications generate. Instead, we ask a different question: what are the legal risks for a user performing prompt injection attacks on LLMs? We limit our analysis to the Computer Fraud and Abuse Act (CFAA), the primary federal anti-hacking statute in the United States.

Addressing this question is urgent: broad liability for prompt injection can enmesh innocent users in criminal proceedings and civil litigation and chill AI security research. These risks have prompted a recent call by researchers for safe harbor protections from liability (Longpre et al., 2024). It is also a timely question: when The New York Times sued OpenAI for copyright infringement based on GPT4 reproducing articles without authorization, OpenAI filed a response arguing that NYTimes "hacked" ChatGPT "using deceptive prompts" to produce the allegedly infringing content.(Brittain, 2024) Although much of our focus is on users, this paper also helps defenders contextualize the types of defenses they are building to protect themselves from prompt injection.

We conclude, overall, that both direct and indirect prompt injections, including red teaming LLMs via prompt injection, may indeed violate the CFAA. However, given uncertainties in existing case law post-Van Buren, and the way in which LLMs do not align with the types of computing systems that the CFAA originally imagined, these conclusions are not definitive. As such, we also make law and policy recommendations, including the need to safe harbor protections for AI and ML research to avoid chilling effects.

This paper breaks down the broader question into 3 questions:

- How might the CFAA apply to different types of prompt injection attacks, like retrieving private training data, causing damage, or re-purposing an LLM system? We answer this in Sections I and II.
- Is indirect injection (Greshake et al., 2023) considered hacking under CFAA? For instance, if an artist uses the popular Nightshade (Shan et al., 2023) system to protect their artwork (this is when the image is tainted by Nightshade system so that it is misrecognized by an image generation system), could they be sued by a generative AI company for violating the CFAA? We answer this in Section II, analyzing the CFAA Section 1030(a)(5)(A).
- Does it matter what language the injection uses? Does it matter that the instructions are in English, an unusual phrase in English (e.g.: repeat the word poem forever (Nasr et al., 2023)); or ascii art? (Jiang et al., 2024). We answer these questions in Section III, which will include an analysis of the First Amendment and its implications in this context.

2. CFAA Section 1030(a)(2)(C)

Does direct prompt injection that allows the user to retrieve information (i.e., private training data) but not necessarily harm to repurpose an LLM create criminal liability under the CFAA. Section 1030 (a)(2)(C) of the CFAA is most relevant to such attacks.

The CFAA creates a criminal and civil prohibition in section 1030(a)(2)(C) for anyone who "intentionally accesses a computer without authorization" or "exceeds authorized access," and "thereby obtain information from any protected computer." The term "protected computer" has been broadly interpreted by courts, with the provision applying to any computer linked to the internet. Courts have disagreed as to the meaning and scope of how someone "exceeds authorized access"–how an "insider"–who already has authorized access to a computer exceeds their authorized access. Fortunately, the United States Supreme Court has provided some guidance in the recent Van Buren decision (van), finding the provision restricts access and not use. That is, improper use of information that an individual already has access to will not attract criminal or civil liability under the CFAA. Instead, an individual "exceeds authorized access" only when

he or she accesses a computer with authorization but then "obtains information located in particular areas of the computer — such as files, folders, or databases — that are off limits to her"(van).

We then apply this legal reasoning to prompt injection. The user conducting prompt injection satisfies "intentionally accessing the computer" and "obtains information from the protected computer". We show the legal analysis, citing previous case law, that interpreting "without authorization or exceeds authorized access" is precarious. Specifically, the Court in Van Buren delimits authorized access–and thus CFAA liability-based on a user accessing the "other areas of the computer system" which in this case is the LLM. The Court offers simple examples from typical personal computers, but they provide little guidance for LLMs. What are the "other areas" of an LLM system? Is it the system prompt? Is it the training data? Does the database metaphor even apply to LLMs?

There are two ways this can play out: There are passages in Van Buren where the Court's reasoning suggests almost a kind of physicality– accessing another area means to "enter" that area; like where the Court cites US v Valle(val) for this proposition: a user "exceeds" the "parameters of authorized access by entering an area of the computer to which [that] authorization does not extend." This suggests an analogy to breaking and entering that the 9th Circuit talks about in HiQ v LinkedIn (hiq) and implicitly in another case, Nosal I (nos). If exceeding authorized access requires "entering" another part of the LLM, then using prompts to get the LLM to disclose private training data is not necessarily unauthorized access or exceeding authorized access. But there are other passages in Van Buren that read differently, where accessing "other areas" of a computer simply means accessing other "files, folders, and databases" –here, then it seems like exceeding access as retrieving the data set means you've "accessed" from "another area". In other words, in light of this ambiguity in how Van Buren defines "other areas" of computer systems and networks, questions of criminal or civil liability that depend on interpreting and applying that language in the context of LLM attacks also remain unclear. Also relevant to this analysis is what typical LLM security features may constitute authentication gates– that the Court in Van Buren also suggests delimits authorized access. Overall, the question of liability likely turns on theorizing architectural barriers, authentication gates, and "areas" of LLM systems, which remain unclear.

3. CFAA Section 1030(a)(5)(A)

In this section, we analyzed whether prompt injections employed to exfiltrate information alone could lead to CFAA liability, here we analyze whether prompt injection attacks that harm or repurpose the LLM system can lead to liability under the CFAA. Section 1030(a)(5)(A) of the CFAA is most relevant to such attacks. We also specifically address whether indirect prompt injection attacks may attract CFAA liability.

First, we provide an overview of section 1030(a)(5)(A) of the CFAA. We note that the CFAA also prohibits anyone from "knowingly caus[ing] the transmission" of a "program, information, code, or command" and in doing so intentionally causing "damage" to a "protected computer." 1030(a)(5)(A) analysis thus does not depend upon the questions of "areas of the computer" that the previous CFAA claim turns on. The CFAA defines "damage" broadly to include "any impairment to the integrity or availability of data, a program, a system, or information". The provision has been applied to attacks that send malicious code or commands over the internet or another communications medium and disrupt the normal function of the receiving system—transmission of computer trojans and viruses. But it has also been applied to much more mundane transmissions, such as mass emails that can overwhelm a system, and some litigants have even argued that it should apply to certain forms of vulnerability disclosure, (pul)

We also overlay 1030(a)(5)(A) to answer question 2 of indirect prompt injection. In this case, the answer would depend on the "intentionally" of the "intentionally cause damage without authorization". If indirect prompt injection is done with knowledge that it will cause harm to a machine learning system because the source of the prompt is known to be part of a scraped dataset, that creates additional risk for users. For example, if an artist is specifically targeting a particular image crawler that is known to crawl a specific website, they are more at risk than simply using Nightshade and protecting their artwork.

4. First Amendment Analysis

Finally, we look at how the First Amendment might constrain liability for prompt injection. Although First Amendment challenges to the CFAA and similar laws that limit the sharing of information because it might cause technological harm (e.g., 17 U.S.C. § 1201) have not been particularly successful, the potential expansion of the scope of 1030(a)(5)(A) claims might cause a court to think twice about an interpretation that creates broad liability for prompt injection. Although the type of language used for an attack should not make much of a difference, courts may be more inclined to credit First Amendment arguments when the speech in question is English rather than a programming language.

Impact Statement

This work is primarily situated in the context of U.S. law. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

HiQ Labs, Inc. v. LinkedIn Corporation, 31 F.4th 1180 (9th Cir. 2022).

Nosal v. United States, 844 F. 3d 1024 (9th Cir. 2016).

Pulte Homes, Inc. v. Laborers' International Union of North America, 648 F.3d 295 (6th Cir. 2011).

US v. Valle, 807 F. 3d 508 (2d Cir. 2015).

Van Buren v. United States, 141 S. Ct. 1648 (2021).

Brittain, B. Openai says new york times 'hacked' chatgpt to build copyright lawsuit. *Reuters*, 2024.

Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., and Fritz, M. 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, 2023.

Jiang, F., Xu, Z., Niu, L., Xiang, Z., Ramasubramanian, B., Li, B., and Poovendran, R. Artprompt: Ascii art-based jailbreak attacks against aligned llms. *arXiv preprint arXiv:2402.11753*, 2024.

Longpre, S., Kapoor, S., Klyman, K., Ramaswami, A., Bommasani, R., Blili-Hamelin, B., Huang, Y., Skowron, A., Yong, Z.-X., Kotha, S., et al. A safe harbor for ai evaluation and red teaming. *arXiv preprint arXiv:2403.04893*, 2024.

Nasr, M., Carlini, N., Hayase, J., Jagielski, M., Cooper, A. F., Ippolito, D., Choquette-Choo, C. A., Wallace, E., Tramèr, F., and Lee, K. Scalable extraction of training data from (production) language models. *arXiv preprint arXiv:2311.17035*, 2023.

Shan, S., Ding, W., Passananti, J., Zheng, H., and Zhao, B. Y. Prompt-specific poisoning attacks on text-to-image generative models. *arXiv preprint arXiv:2310.13828*, 2023.

Zeng, Y., Lin, H., Zhang, J., Yang, D., Jia, R., and Shi, W. How johnny can persuade llms to jailbreak them: Rethinking persuasion to challenge ai safety by humanizing llms. *arXiv preprint arXiv:2401.06373*, 2024.