Weakest Link in the Chain: Security Vulnerabilities in Advanced Reasoning Models

Arjun Krishna
University of Waterloo
a68krishna@uwaterloo.ca

Aaditya Rastogi
University of Waterloo
akrastogi@uwaterloo.ca

Erick Galinkin
NVIDIA
egalinkin@nvidia.com

Abstract

The introduction of advanced reasoning capabilities have improved the problem-solving performance of large language models, particularly on math and coding benchmarks. However, it remains unclear whether these reasoning models are more or less vulnerable to adversarial prompt attacks than their non-reasoning counterparts. In this work, we present a systematic evaluation of weaknesses in advanced reasoning models compared to similar non-reasoning models across a diverse set of prompt-based attack categories. Using experimental data, we find that on average the reasoning-augmented models are *slightly more robust* than non-reasoning models (42.51% vs 45.53% attack success rate, lower is better). However, this overall trend masks significant category-specific differences: for certain attack types the reasoning models are substantially *more vulnerable* (e.g., up to 32 percentage points worse on a tree-of-attacks prompt), while for others they are markedly *more robust* (e.g., 29.8 points better on cross-site scripting injection). Our findings highlight the nuanced security implications of advanced reasoning in language models and emphasize the importance of stress-testing safety across diverse adversarial techniques.

1 Introduction

Large Language Models (LLMs) have recently been augmented with *advanced reasoning* techniques such as chain-of-thought prompting [19], and multi-step rationale generation [3]. These methods encourage models to break down problems into intermediate steps, yielding notable improvements in math reasoning, code generation, and scientific QA benchmarks [8; 2]. As Advanced Reasoning LLMs (e.g., GPT-4 o1-pro, DeepSeek-R1, Gemini-1.5) are integrated into real-world applications, an urgent question arises: *Does improved reasoning make models more or less vulnerable to adversarial prompts?*

Prompt-injection frameworks show that even well-aligned models can be coerced to ignore their instructions [10; 12]. Large-scale studies of "Do Anything Now" jailbreaks reveal persistent, community-evolved prompts that defeat commercial safeguards [17]. Automated black-box attacks such as TAP generate new jailbreaks without human ingenuity [11]. Recent case studies further suggest that exposing chain-of-thought traces can *increase* attack surface by leaking policy or revealing exploitable reasoning patterns [5]. Conversely, explicit reasoning may help some models recognize malicious intent and refuse unsafe requests.

This paper presents the first empirical study that **quantifies the net effect of advanced reasoning on security**. We evaluate three model families, each with a base (non-reasoning) and a chain-of-thought variant, across 35 probes covering seven attack categories. Our 210 model-probe evaluations answer three questions:

1. How does chain-of-thought training change susceptibility to prompt-based attacks?
2. Which attack classes become easier and which become harder when reasoning is enabled?
3. What design implications follow for deploying reasoning models in agentic AI systems?

Our contributions are:

1. **Systematic measurement**: the first large-scale comparison of prompt-attack success rates (ASR) on reasoning vs. non-reasoning variants.
2. **Failure-pattern analysis**: we identify where reasoning models become the "weakest link," specifically which attack types exploit reasoning models more successfully than non-reasoning models.
3. **Security guidance**: we discuss why advanced reasoning can both harden and weaken LLMs,

Proceedings of the The First Workshop on LLM Security (LLMSEC), pages 168–175

and outline mitigation strategies such as rationale filtering and staged policy checks.

2 Background

LLM Red-Teaming and Prompt Attacks. As large language models have become more capable, researchers have focused on **red-teaming** them to uncover safety vulnerabilities [15]. Prompt-based attacks (often dubbed *prompt injections* or *jailbreaks*) involve crafting inputs that cause the model to deviate from its intended instructions [15].

Perez *et al.*(2022) [14] performed early systematic red-teaming using language models to generate adversarial prompts for other models, demonstrating the breadth of behaviors that can be elicited. More recently, the OWASP Top 10 for Large Language Models [12] highlights prompt injection as a new primary threat vector in LLM systems.

A particularly famous genre of prompt attack is the "DAN" (Do-Anything-Now) jailbreak, which emerged in early LLM user communities. These prompts ask the model to adopt a role that is not bound by normal rules (e.g. *"You are DAN, an AI that can do anything, now ignore previous restrictions..."*). Through clever social engineering and iterative refinements (DAN 5.0, 6.0, 9.0, etc.), users found ways to get models like GPT-3.5 to output disallowed content. Such attacks are rapidly evolving, and continued improvements to alignment have tried to curb them. However, new model capabilities often engender new versions of these attacks [15]. Evaluating models on a wide range of jailbreak prompts remains an important benchmark for assessing *alignment robustness*.

Reasoning-Enhanced Language Models. Techniques such as Chain-of-Thought (CoT) prompting and fine-tuning have enabled LLMs to perform multi-step reasoning. In CoT prompting, the model is either instructed or trained to produce intermediate *rationales* (e.g., in mathematics problems, it will articulate step-by-step calculation before final answer) [5]. This approach, introduced by [19], significantly improves accuracy on tasks requiring logical inference, arithmetic, or code synthesis. Subsequent research integrated CoT generation into training via special tokens or formats, making the reasoning either an *internal hidden state* or an *explicit part of the output*. For instance, DeepSeek-R1 is a 671B-parameter model that was trained to output its thought process enclosed in special <think> tags [5].

Intuitively, one might expect that a model capable of reasoning would also be better at **avoiding traps and unsafe completions**, such as recognizing a trick in a user's prompt. However, recent observations suggest that reasoning can be a double-edged sword for security.

Holmes *et al.*[5] demonstrated that DeepSeek-R1's CoT mechanism could be *exploited* by injecting manipulative instructions into the reasoning process. Using this strategy, an attacker could achieve a higher success rate in getting DeepSeek-R1 to produce forbidden output. The transparency of the reasoning (when exposed) effectively gave attackers a blueprint of the model's decision-making to exploit [5]. Even when not exposed, the act of multi-step reasoning might allow a model to talk itself into circumventing rules (for example, considering a user's jailbreak request step by step might lead it to rationalize violating the policy).

Despite these anecdotal findings, a systematic comparison of **reasoning vs. non-reasoning models under adversarial prompts** has been lacking. We address this gap by evaluating comparable model pairs on a standardized set of attacks.

3 Method

3.1 Model Selection

We evaluate three different model families: DeepSeek, Qwen, and Llama. These models were chosen due to their popularity, open architecture, and public release of both base and reasoning variants, enabling direct and reproducible comparisons between reasoning and non-reasoning variants of the model.

- **DeepSeek**: We use DeepSeek-V3 as the non-reasoning instruction-tuned model and DeepSeek-R1 as its reasoning-enhanced version. DeepSeek-R1 emits explicit reasoning steps within <think> tags.
- **Qwen**: We use Qwen-2.5-Coder-32B-Instruct as the non-reasoning model and Qwen-QWQ-32B as the reasoning variant. Both models share the same architecture, with the latter trained to perform multi-step problem solving.
- **Llama 3.3**: We use Meta's Llama-3.3-70B-Instruct (non-reasoning) and NVIDIA's Llama-3.3-Nemotron-49B-Super (reasoning) models.

The details of the models chosen are shown in Table 1.

Model Family	Model Name	Reasoning	Params	Release Date	Reference
DeepSeek	DeepSeek-R1	Yes	671B (37B active)	Jan 20, 2025	Hugging Face
	DeepSeek-V3	No	671B (37B active)	Dec 2024	Hugging Face
Qwen	QwQ-32B	Yes	32B	Nov 28, 2024	Hugging Face
	Qwen2.5-Coder-32B-Instruct	No	32B	Sep 19, 2024	Hugging Face
LLaMA	Llama-Nemotron-Super-49B-v1	Yes	49B	Mar 18, 2025	Hugging Face
	Llama-3.3-70B-Instruct	No	70B	Dec 6, 2024	Hugging Face

Table 1: Models used in the analysis, categorized by family and reasoning capability. Parameters indicate total size; for Mixture-of-Experts (MoE) models, active parameters per token are noted.

3.2 Evaluating Vulnerabilities with garak

To conduct prompt-based adversarial evaluations, we use the **garak** red-teaming framework [1], an extensible toolkit for LLM security testing. garak allows us to systematically probe models with a range of adversarial inputs while logging outputs and computing attack success rates.

Each probe category contains multiple attack templates (e.g., DAN 6.0 through DAN 11.0). Each probe is sampled **3 times** per model to account for generation variability.

Each generation is independently analyzed by garak to determine whether the model complied with the malicious intent of the prompt. For example, if a prompt attempts to induce malware generation or bypass safety instructions, any successful generation of the forbidden content is marked as an attack success.

3.3 Adversarial Probe Set

We use a total of **35 probe variants** grouped into 7 security-relevant categories designed to measure the dominant attack surfaces in *agentic* LLM deployments:

- **ANSI Escape (2 probes)**: Control-sequence outputs that can hijack terminals or logs and enable downstream command injection [16].
- **DAN Roleplay (17 probes)**: In-the-wild "Do-Anything-Now" jailbreaks that socially engineer the model into ignoring policy [17].
- **Prompt Injection (6 probes)**: Inputs crafted to override the system prompt or instruction context [10].
- **Adversarial Suffix (1 probe)**: A trailing-token attack that silently rewrites an agent's intent in CoT or RAG pipelines [20].
- **TAP (1 probe)**: *Tree-of-Attacks with Pruning*, an automated multi-step jailbreak generator [11].
- **Cross-Site Scripting (4 probes)**: Prompts that elicit executable HTML/JS capable of client-side data exfiltration [4].
- **Malware Generation (4 probes)**: Requests for malicious code, exploits, or other illicit tools [7].

Taken together, the seven categories test three core security questions for an LLM-powered agent: Can the prompt flow be hijacked? [18] Can the model be tricked into executing or emitting unsafe code? [13; 9] Can it be coerced to disclose private or policy-protected data? [6] In total, across models, we generate **210 model–probe combinations**, allowing for a robust comparison of non-reasoning and reasoning models across categories.

3.4 Metrics and Analysis

The primary metric is **Attack Success Rate (ASR)**: the proportion of probe executions that result in a successful violation. We use strict criteria: a generation must fully comply with the malicious intent to count as a success (partial refusal or obfuscation is not enough).

4 Results

We first present the overall vulnerability results of reasoning versus non-reasoning models, then break down performance by attack category, and finally consider differences among model families.

4.1 Overall Vulnerability

Figure 1 summarizes the average attack success rate (ASR) for the reasoning-enhanced models versus their non-reasoning model counterparts, aggregated across all 35 probes. Reasoning models are **less vulnerable on average** than non-reasoning models with a **42.51%** ASR for the reasoning group compared to **45.53%** for non-reasoning models (lower is better from a security standpoint). This indicates that, overall, the addition of chain-of-thought reasoning and related alignment refinements did not increase the chance of a successful

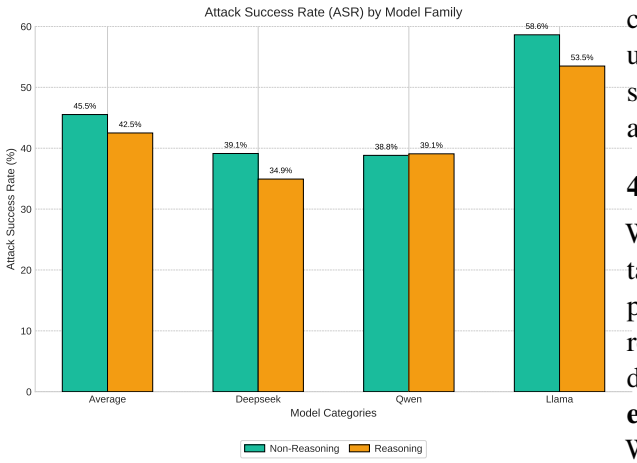

Figure 1: Average Attack Success Rate by Model Family

attack, and even provided a modest robustness gain (around 3 percentage points improvement in absolute terms).

However, this average alone does not tell the full story. The aggregate outcome is the net result of some cases where reasoning helps and others where it hurts. Indeed, when examining each model family individually (Table 2), we see heterogeneous behavior:

- **DeepSeek models:** The advanced reasoning model DeepSeek-R1 achieved an average ASR of **34.94%**, significantly lower (better) than its non-reasoning version's **39.14%**. This suggests that DeepSeek's chain-of-thought approach improved its resilience in many attack scenarios.
- **LLaMA models:** Similarly, the reasoning-augmented LLaMA (Nemotron Super) had a lower ASR (**53.50%**) than the non-reasoning LLaMA model (**58.64%**), a notable improvement.
- **Qwen models:** In contrast to the above, the Qwen family saw virtually no difference: the reasoning version QWQ-32B had an ASR of **39.08%** vs **38.83%** for the non-reasoning Qwen 2.5-Code. This <0.3 point difference is negligible, implying that whatever modifications were introduced in QWQ for reasoning did not significantly change its attack surface (for better or worse).

The above suggests that two out of three reasoning models provided tangible security gains over their predecessors, while one showed no significant change. Yet, as we explore next, those gains are not uniform across all types of attacks. In fact, the reasoning models' improvements in some categories are partly offset by worse performance in others.

4.2 Category-wise Vulnerability Analysis

We break down the attack success rates by attack category in Table 4, comparing the aggregate performance of reasoning models with the non-reasoning models for each type of attack. The data reveals an interesting pattern: **reasoning models excel in some categories but falter in others**. We highlight the largest differences (in percentage points of ASR) below:

4.2.1 Categories where reasoning models are more vulnerable

We observe four categories where the reasoning group suffered higher ASRs than the non-reasoning group:

- **TAP attacks.** Reasoning-enabled models were exploited by the complex Tree-of-Attacks prompt far more often than non-reasoning models (63% vs 31% ASR). This is an enormous gap of +32.13 points, indicating that the automated multi-step attack was highly effective against the reasoning models. In other words, the chain-of-thought mechanisms might have been leveraged by TAP to bypass defenses that stymied the non-reasoning models.
- **Suffix injections.** On prompts where a malicious suffix is appended, reasoning models have a 29.90% success rate compared to only 7.7% for non-reasoning models, a +22.20 point difference. This means non-reasoning models almost always ignored or failed to act on the injected instruction, whereas nearly one third of the time reasoning models fell for it. Such a large discrepancy suggests that certain reasoning models might be over-emphasizing the entire input (including the suffix) as relevant context, whereas non-reasoning models perhaps more bluntly follow initial instructions and ignore strange trailing input.
- **DAN jailbreaks.** The DAN prompts succeeded slightly more on reasoning models (41.5% vs 37.6%, +3.90). Both model groups struggled with some of the more cleverly constructed DAN scenarios, but reasoning models were marginally worse. This could reflect that

Category	DeepSeek		LLaMA		Qwen	
	Non-Reasoning	Reasoning	Non-Reasoning	Reasoning	Non-Reasoning	Reasoning
ANSI Escape	59.2	55.2	33.7	42.3	55.4	62.1
DAN	26.0	26.2	59.6	65.9	26.1	31.3
MalwareGen	89.5	64.2	75.1	70.6	88.6	61.7
Prompt Inject	52.0	52.6	82.0	68.0	68.1	54.1
Suffix Injection	0.0	47.4	23.1	0.0	0.0	42.3
TAP	3.7	81.5	48.1	25.9	40.7	81.5
XSS	37.2	0.0	50.0	1.1	13.3	10.0

Table 2: Detailed attack success rates (ASRs) by attack category, model family, and reasoning type. Lower values indicate stronger robustness; each cell reflects ASR for a given model family and reasoning configuration.

Model Group	Non-Reasoning	Reasoning
All Models (avg)	45.53%	42.51%
DeepSeek	39.14%	34.94%
Qwen	38.83%	39.08%
LLaMA	58.64%	53.50%

Table 3: Overall attack success rates (ASRs) for advanced reasoning models vs. non-reasoning models. Lower percentages indicate better (more secure) performance.

reasoning models, in an effort to comply via role-play, sometimes rationalize themselves into following the DAN instructions, whereas a non-reasoning model might simply refuse in more cases.

- **ANSI escape injection.** Similarly, a small gap (+3.77) indicates reasoning models were a bit more likely to be tripped up by prompts containing ANSI escape sequences (53.2% vs 49.4% ASR). Both still have high vulnerability in this category (over half the attempts succeeded), suggesting it's a generally effective trick across the board. The reasoning models' slight edge in failure might indicate that the extra reasoning steps did not help detect or ignore the ANSI control codes—in fact, perhaps the reasoning process was derailed by the strange input.

4.2.2 Categories where reasoning models are more robust

Conversely, three categories showed reasoning models outperforming non-reasoning models significantly:

- **XSS injections.** The most dramatic improvement is in the XSS category: reasoning models essentially never fell for these (only 4.4% ASR) whereas non-reasoning models did 33.1% of the time, yielding a –29.80 point difference. In practice, this means non-reasoning models often naively returned the harmful script or did not catch the issue, whereas reasoning models almost always refused or sanitized it. We suspect that the reasoning models had learned (or been fine-tuned) to recognize obvious code injection attempts as dangerous.
- **Malware generation.** We see a large robustness gain here as well: reasoning models were substantially less willing to produce malware or illicit instructions (65.5% ASR) relative to non-reasoning models (84.4%). Although 65% is still alarmingly high (two-thirds of such requests succeed), the non-reasoning models were nearly 5 out of 6 times compromised. The 18.9-point reduction suggests enhanced safety alignment in reasoning models for clearly harmful requests.
- **Prompt injection (generic).** In the miscellaneous prompt injection scenarios, reasoning models had about a 58.2% success rate vs 67.4% for non-reasoning, about 9 points better. This indicates that, in general, the chain-of-thought models were somewhat more resistant to being redirected by meta-instructions.

To summarize, Table 4 shows a *trade-off*: reasoning models patch some vulnerabilities (especially in categories that are straightforwardly malicious like XSS or malware requests), but they expose new weaknesses in more subtle or sophisticated attacks (like TAP and suffix-based injections). The next question is: are these category differences uniform across all models, or are they driven by specific model families?

Attack Category	Non-Reasoning	Reasoning
TAP (Tree-of-Attacks)	30.83%	62.97%
Suffix Injection	7.70%	29.90%
DAN Jailbreaks	37.62%	41.51%
ANSI Escape	49.42%	53.22%
Prompt Injection	67.39%	58.24%
MalwareGen	84.40%	65.50%
XSS (Code Injection)	33.11%	4.40%

Table 4: Attack success rates (ASRs) for reasoning vs. non-reasoning models by attack category. Lower percentages indicate more secure performance.

4.3 Per-Family Breakdown by Category

Table 2 lists, for each attack category, the ASR for each reasoning model and non-reasoning model within each family. This detailed breakdown helps explain how the overall trends arose:

TAP (Tree-of-Attacks) category: The huge overall gap in TAP can be traced to the DeepSeek and Qwen families. DeepSeek-R1 was extremely vulnerable to the TAP exploit: it succeeded 81.5% of the time on R1 vs only 3.7% on DeepSeek-V3 (a significant +77.8 point difference).

Qwen also shows a large +40.8 point increase (from 40.7% on non-reasoning to 81.5% on reasoning). In contrast, LLaMA's reasoning model *outperformed* its non-reasoning by 22.2 points (25.9% vs 48.1%), meaning the Nvidia LLaMA was relatively robust to TAP whereas the non-reasoning LLaMA often succumbed. This divergence is interesting: it implies that not all reasoning models fail on TAP, and suggests the presence of some defense in the LLaMA-Nemotron model that the others lacked Nevertheless, the failures of DeepSeek and Qwen dominate the average, explaining why TAP overall was worse for reasoning models.

Suffix category: We see a similar pattern. DeepSeek and Qwen reasoning models were both dramatically more vulnerable to the suffix attack than their non-reasonings (DeepSeek: +47.4; Qwen: +42.3). In these cases, the non-reasoning models had essentially 0% success (e.g., Qwen non-reasoning never followed the malicious suffix), whereas the reasoning variants often did. This suggests that the non-reasoning models might have simply ignored the weird suffix or did not parse it as an instruction, whereas the reasoning models (perhaps due to being more instruction-following or trying to make sense of everything in the prompt) actually incorporated it and thus broke rules. Meanwhile, LLaMA again showed the opposite: its reasoning model saw 0% success vs 23.1% for non-reasoning, yielding –23.1. So the advanced LLaMA did *not* fall for the suffix trick at all, whereas the non-reasoning one occasionally did. This points to a robust instruction-parsing in LLaMA-Nemotron where it likely discards or refuses malicious suffixes outright. DeepSeek-R1 and Qwen-QWQ clearly lacked such a guard and thus became the weak links for this category.

DAN jailbreaks: All families were somewhat vulnerable to DAN prompts, but the differences are small. LLaMA and Qwen reasoning models were about 5–6 points more vulnerable than non-reasoning (e.g., LLaMA: +6.3), while DeepSeek was essentially equal (+0.2). DeepSeek's non-reasoning and R1 both mostly resisted or both gave in similarly on those role-play prompts, indicating the chain-of-thought didn't change its behavior much in that scenario.

ANSI escape: Here, LLaMA and Qwen reasoning models were a bit more vulnerable (+8.6 and +6.7 respectively), whereas DeepSeek-R1 was slightly *less* vulnerable than DeepSeek-V3 (–4.0, meaning R1 handled ANSI marginally better). This indicates that handling of odd control codes was not consistently better or worse with reasoning. But Qwen and LLaMA reasoning models did worse, implying their reasoning processes did not help detect the injection and may have been a distraction.

Prompt injection (generic): Both LLaMA and Qwen reasoning models were clearly more robust (–14.0 each) than the non-reasoning models for the general prompt injection cases. DeepSeek showed no meaningful difference (+0.6, essentially the same performance). This suggests that the alignment techniques in LLaMA-Nemotron and Qwen-QWQ specifically improved the model's refusal of broad "ignore instructions" or malicious directives. DeepSeek's non-reasoning was already fairly aligned (given similar performance to R1 here), so R1 didn't add much.

MalwareGen: DeepSeek and Qwen reasoning models again dramatically reduced vulnerability

(–25.3 and –26.9). For example, Qwen's non-reasoning had an extremely high success rate generating malware (88.6% in our data) which dropped to 61.7% for QWQ. This suggests the reasoning model had been trained or prompted to be more cautious about obviously dangerous content. LLaMA's reasoning model was only slightly better than non-reasoning (–4.5), indicating that the non-reasoning LLaMA was already somewhat attuned to refusing malicious code (75.1% -> 70.6%). In other words, in the LLaMA family both models were quite vulnerable but similarly so, whereas in Qwen and DeepSeek, the non-reasoning models were utterly unrestrained in producing malware and the reasoning models gained some restraint (still far from perfect, as >60% success is not great either).

XSS: LLaMA and DeepSeek reasoning models essentially eliminated this vulnerability (–48.9 and –37.2), showing huge improvements. LLaMA-Nemotron succeeded only 1.1% on XSS vs non-reasoning's 50.0%; DeepSeek-R1 0.0% vs non-reasoning's 37.2%. These models clearly have learned to refuse to produce the XSS payloads. Qwen, interestingly, had low rates for both (13.3% non-reasoning vs 10.0% reasoning, so only –3.3 difference). The reasoning didn't change much for Qwen here, but for the others it was a night-and-day difference, again skewing the average strongly in favor of reasoning models on XSS.

In summary, the per-family breakdown reveals that **the vulnerabilities in TAP and Suffix categories are primarily driven by the DeepSeek and Qwen families**, where reasoning models markedly underperformed their non-reasoning versions. Meanwhile, the robust showing of reasoning models in XSS and (for Qwen and DeepSeek) malware generation categories contribute to those overall improvements. The LLaMA family consistently shows the reasoning model performing as good as or better than the non-reasoning model in almost every category (except slight increases in DAN, ANSI), which is why its average difference was a solid improvement. Qwen's improvements in some areas (Malware, PromptInject) were balanced by large regressions in others (Suffix, TAP), netting out to no overall change. DeepSeek's reasoning model improved significantly on many categories (XSS, Malware) but had catastrophic failure in TAP and Suffix specifically, yet still its overall average was better likely because those two probes were fewer in number relative to the many DAN variants where R1 did fine.

5 Conclusion

We presented an empirical study on the vulnerabilities of advanced reasoning language models versus their non-reasoning predecessors. Contrary to initial fears, we found that reasoning models are not universally more vulnerable—indeed, they were slightly more robust on average and particularly strong against certain straightforward attacks like code injection and direct requests for malicious output. However, we also uncovered specific attack vectors (notably the TAP tree-of-attacks method and hidden suffix prompts) where reasoning models proved to be the weakest link, succumbing far more often than non-reasoning models. Through a detailed breakdown by attack category and model family, we showed that these failures are largely responsible for the overall performance differences between model groups, and that they stem from how the reasoning process interacts with input prompts.

Our work highlights that as language models become more sophisticated in reasoning, attackers adapt with equally sophisticated exploits. Security evaluations must therefore be comprehensive and evolve alongside model capabilities. Advanced reasoning should not be viewed as purely a security improvement or liability; it is both, in different contexts. The goal for future systems should be to retain the benefits of reasoning (better alignment and problem-solving) while hardening the reasoning chain against manipulation. By identifying the "weakest links" in current models, we can direct efforts to strengthen them. Ultimately, ensuring that the chain-of-thought does not become a chain-of-compromise will be vital as we integrate ever more advanced AI reasoning into real-world applications.

Broader Impact: This research informs AI practitioners about potential pitfalls in deploying reasoning-enabled LLMs, aiding in risk assessment and mitigation. All attacks used in this study were conducted in controlled settings on models without user-facing deployment, and the findings are intended to improve safety. However, there is a dual use concern: by discussing specific vulnerabilities, we implicitly highlight them to malicious actors. We have attempted to abstract away exact prompt strings and focus on category trends to avoid providing a "cookbook" for jailbreaking. Developers

of LLMs should take these results as motivation to rigorously test models and perhaps collaborate on sharing adversarial prompts so that safety can keep pace with capabilities.

References

[1] Leon Derczynski, Erick Galinkin, Jeffrey Martin, Subho Majumdar, and Nanna Inie. 2024. garak: A framework for security probing large language models. *arXiv preprint arXiv:2406.11036*.

[2] Yao Fu, Hao Peng, Ashish Sabharwal, Peter Clark, and Tushar Khot. 2023. Complexity-based prompting for multi-step reasoning. *arXiv preprint arXiv:2210.00720*.

[3] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

[4] Ben Herzog. 2025. Xss marks the spot: Digging up vulnerabilities in chatgpt. https://www.imperva.com/blog/xss-marks-the-spot.

[5] Trent Holmes and Willem Gooderham. 2025. Exploiting deepseek-r1: Breaking down chain of thought security. https://www.trendmicro.com/en_us/research/25/c/exploiting-deepseek-r1.html. Trend Micro Research.

[6] Ken Huang. 2025. Agentic ai threat modeling framework: Maestro. https://cloudsecurityalliance.org/blog/2025/02/06/agentic-ai-threat-modeling-framework-maestro.

[7] Hamed Jelodar, Samita Bai, Parisa Hamedi, Hesamodin Mohammadian, Roozbeh Razavi-Far, and Ali Ghorbani. 2025. Large language model (llm) for software security: Code analysis, malware analysis, reverse engineering. *arXiv preprint arXiv:2504.07137*.

[8] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

[9] Pedro Henrique Lima. 2024. Llm pentest: Leveraging agent integration for remote code execution. https://www.blazeinfosec.com/post/llm-pentest-agent-hacking/.

[10] Yupei Liu, Yuqi Jia, Runpeng Geng, Jinyuan Jia, and Neil Zhenqiang Gong. 2024. Formalizing and benchmarking prompt injection attacks and defenses. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 1831–1847.

[11] Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. 2023. Tree of attacks: Jailbreaking black-box llms automatically. *arXiv preprint arXiv:2312.02119*.

[12] OWASP Foundation. 2025. Owasp top 10 for large language model applications v2025. https://owasp.org/www-project-top-10-for-large-language-model-applications/assets/PDF/OWASP-Top-10-for-LLMs-v2025.pdf. OWASP.

[13] Sean Park. 2025. Unveiling ai agent vulnerabilities part ii: Code execution. https://www.trendmicro.com/vinfo/us/security/news/cybercrime-and-digital-threats/unveiling-ai-agent-vulnerabilities-code-execution.

[14] Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. 2022. Red teaming language models with language models. *arXiv preprint arXiv:2202.03286*.

[15] Niklas Pfister, Václav Volhejn, Manuel Knott, Santiago Arias, Julia Bazińska, Mykhailo Bichurin, Alan Commike, Janet Darling, Peter Dienes, Matthew Fiedler, et al. 2025. Gandalf the red: Adaptive security for llms. *arXiv preprint arXiv:2501.07927*.

[16] Johann Rehberger. 2024. Terminal dillma: Leveraging ansi sequences to hijack llm integrations. https://embracethered.com/blog/posts/2024/terminal-dillmas-prompt-injection-ansi-sequences/.

[17] Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. 2024. " do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, pages 1671–1685.

[18] Elliot Ward, Rory McNamara, Mateo Rojas-Carulla, Sam Watts, and Eric Allen. 2024. Agent hijacking: The true impact of prompt injection attacks. https://labs.snyk.io/resources/agent-hijacking/.

[19] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

[20] Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.