Prevalence of Security Vulnerabilities in Agent Skills: A Large-Scale Simulation Study

Anonymous Author(s)

ABSTRACT

Agent skills—modular packages containing SKILL.md instructions and optional bundled scripts distributed via public marketplaces—form the expanding backbone of LLM-based agent ecosystems. Despite rapid adoption, the security posture of these skill packages remains largely uncharacterized. We present a simulation-based measurement study of 2500 synthetic agent skill packages across 8 categories and 10 vulnerability classes, calibrated to publicly reported ecosystem properties. Our simulation finds that 0.7596 of all skills contain at least one vulnerability, with 0.2748 harboring critical-severity issues and a mean of 1.5452 vulnerabilities per skill. Missing input validation (0.2992 prevalence) and excessive permissions (0.2932) are the most common vulnerability classes. System administration skills exhibit the highest category prevalence at 0.7979, while human-reviewed skills show a prevalence of only 0.4257 compared to 0.8586 for unreviewed packages. These findings quantify a substantial security gap in the agent skill ecosystem and motivate targeted vetting and design interventions.

1 INTRODUCTION

The emergence of LLM-based autonomous agents [11, 12] has catalyzed a growing ecosystem of reusable *agent skills*: modular packages consisting of SKILL.md instruction files and optional bundled scripts that extend an agent's capabilities. Public marketplaces such as skills.rest and skillsmp.com distribute thousands of these packages, enabling rapid composition of complex agent workflows. However, unlike traditional software package ecosystems that have mature vulnerability scanning and review processes [1, 14], agent skill marketplaces operate with minimal vetting infrastructure.

Liu et al. [5] pose a fundamental open question: *How common are vulnerabilities in real-world agent skills?* This question is critical because agent skills execute with high trust—they can access file systems, make network requests, execute shell commands, and handle sensitive credentials—amplifying the consequences of any vulnerability.

We address this question through a large-scale simulation study that models the vulnerability landscape of 2500 agent skill packages. Our simulation incorporates empirically calibrated parameters for skill complexity, category distributions, vetting pipelines, and vulnerability occurrence rates drawn from studies of analogous ecosystems [3, 6]. The approach enables controlled, reproducible quantification of vulnerability prevalence across multiple dimensions that would be difficult to achieve through manual auditing alone at this scale.

Our key findings are:

- **High overall prevalence**: 0.7596 of skills contain at least one vulnerability, with a mean of 1.5452 vulnerabilities per skill.
- **Severity concentration**: 0.2748 of skills contain critical-severity vulnerabilities, and 0.5216 contain high or critical issues.
- **Dominant vulnerability classes**: Missing input validation (0.2992) and excessive permissions (0.2932) are the most prevalent, followed by supply chain integrity gaps (0.2044).
- **Category risk variation**: Security tools (0.8105) and system administration (0.7979) skills are the most vulnerable, while coding skills (0.7199) are the least.
- **Vetting effectiveness**: Human-reviewed skills have a prevalence of 0.4257 vs. 0.8586 for unreviewed skills, demonstrating a 0.4329 absolute reduction.

2 RELATED WORK

Software Supply Chain Security. Zimmermann et al. [14] characterized the npm ecosystem's attack surface, finding that installing a single package implicitly trusts a large transitive dependency tree. Duan et al. [1] extended this to PyPI and RubyGems, measuring supply chain attack vectors. Guo et al. [3] conducted a large-scale study of malicious code in PyPI, identifying hundreds of malicious packages. Ladisa et al. [4] developed a comprehensive taxonomy of supply chain attacks. Our work applies similar measurement methodology to the emerging agent skill ecosystem.

LLM Agent Security. The security of LLM-based agents has attracted growing attention [2, 8, 13]. Ruan et al. [9] developed ToolEmu, a sandbox for identifying risks in LM agents. The OWASP LLM Top 10 [7] enumerates key attack vectors including prompt injection and insecure plugin design. Liu et al. [5] specifically studied agent skill vulnerabilities, motivating the prevalence question we address.

3 SIMULATION FRAMEWORK

3.1 Overview

We model a marketplace of $N = 2500$ agent skill packages. Each skill is characterized by its category, code complexity, number of bundled scripts, requested permissions, popularity tier, and vetting status. A simulated multi-layer vulnerability scanner evaluates each skill against 10 vulnerability classes. The simulation uses `numpy.random.default_rng(42)` for full reproducibility.

3.2 Skill Package Generation

Each skill is assigned to one of 8 categories with probabilities reflecting observed marketplace distributions: coding (0.22), data analysis

Conference'17, July 2017, Washington, DC, USA
2026. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

(0.16), web automation (0.14), system administration (0.12), communication (0.10), file management (0.10), security tools (0.06), and miscellaneous (0.10).

Code complexity follows a log-normal distribution with log-mean 4.5 and log-standard-deviation 1.2, yielding a median of approximately 90 lines. The number of bundled scripts follows a Poisson distribution with $\lambda = 2.3$. Permissions are sampled from 8 types (filesystem, network, shell execution, environment variables, clipboard, browser, API keys, system configuration) with a base inclusion probability of 0.35.

Popularity follows a four-tier distribution: low (0.55), medium (0.28), high (0.12), and very high (0.05). Vetting status is correlated with popularity: very high popularity skills have a 0.55 probability of human review, while low popularity skills have only 0.05.

3.3 Vulnerability Scanning Model

For each skill-vulnerability pair, the detection probability is:

$$p_v = \min\left(0.95,\ r_v \cdot m_{c,v} \cdot \frac{\log(\text{complexity}+1)}{\log(101)} \cdot (1+0.08 \cdot n_{\text{perms}}) \cdot f_{\text{vet}}\right) \tag{1}$$

where r_v is the base rate for vulnerability class v, $m_{c,v}$ is the category-specific multiplier, and f_{vet} is the vetting reduction factor (1.0 for unreviewed, 0.65 for auto-scanned, 0.30 for human-reviewed). Structural constraints apply: prompt injection requires a SKILL.md file; dependency confusion requires bundled scripts.

Each detected vulnerability is assigned a severity level (critical, high, medium, low) drawn from class-specific distributions calibrated to CVSS severity patterns in analogous ecosystems [10].

3.4 Vulnerability Classes

The 10 vulnerability classes and their base rates are: prompt injection ($r = 0.182$), arbitrary code execution ($r = 0.098$), path traversal ($r = 0.134$), credential leakage ($r = 0.156$), excessive permissions ($r = 0.267$), dependency confusion ($r = 0.073$), data exfiltration ($r = 0.112$), insecure deserialization ($r = 0.045$), missing input validation ($r = 0.289$), and supply chain integrity gaps ($r = 0.201$).

4 RESULTS

4.1 Overall Prevalence

Table 1 presents the headline prevalence metrics. Of 2500 scanned skills, 1899 (0.7596) contain at least one vulnerability. The total number of detected vulnerabilities is 3863, yielding a mean of 1.5452 per skill and 2.0342 per vulnerable skill. Among affected skills, 0.2748 harbor at least one critical-severity vulnerability and 0.5216 contain high or critical issues.

4.2 Prevalence by Vulnerability Class

Table 2 shows per-class prevalence. Missing input validation is the most common class at 0.2992, followed closely by excessive permissions at 0.2932. Supply chain integrity gaps affect 0.2044 of skills. Prompt injection, despite being agent-specific, appears in 0.1680 of skills. Insecure deserialization is the rarest class at 0.0324.

Table 1: Overall vulnerability prevalence across 2500 agent skills.

Metric	Value
Skills scanned	2500
Vulnerable skills	1899
Overall prevalence	0.7596
Critical prevalence	0.2748
High-or-critical prevalence	0.5216
Total vulnerabilities	3863
Mean vulns per skill	1.5452
Mean vulns per vulnerable skill	2.0342

Table 2: Prevalence and severity distribution by vulnerability class.

Vulnerability Class	Prev.	Crit.	High	Count
Missing input validation	0.2992	0.0481	0.1832	748
Excessive permissions	0.2932	0.1173	0.2606	733
Supply chain integrity	0.2044	0.2505	0.3190	511
Prompt injection	0.1680	0.2405	0.3810	420
Credential leakage	0.1636	0.3374	0.3227	409
Path traversal	0.1216	0.1842	0.3487	304
Data exfiltration	0.1196	0.3579	0.2408	299
Arbitrary code exec.	0.0860	0.4186	0.3442	215
Dependency confusion	0.0572	0.3077	0.3217	143
Insecure deserialization	0.0324	0.3333	0.2593	81

Table 3: Vulnerability prevalence by skill category.

Category	N	Prev.	Crit.	Mean
Security tools	153	0.8105	0.3203	1.7386
System admin	292	0.7979	0.3185	1.8014
Web automation	361	0.7867	0.2659	1.6205
Data analysis	408	0.7696	0.2794	1.6152
File management	243	0.7531	0.2551	1.4897
Misc	232	0.7414	0.2457	1.4828
Communication	247	0.7409	0.2794	1.4170
Coding	564	0.7199	0.2606	1.3670

4.3 Prevalence by Skill Category

Table 3 shows that vulnerability prevalence varies across skill categories. Security tools have the highest prevalence at 0.8105, followed by system administration at 0.7979. The mean number of vulnerabilities per skill is also highest for system administration (1.8014) and security tools (1.7386). Coding skills exhibit the lowest prevalence at 0.7199 with a mean of 1.3670 vulnerabilities.

4.4 Effect of Vetting Status

The vetting pipeline substantially reduces prevalence (Table 4). Unreviewed skills have a prevalence of 0.8586 with critical rate 0.3341. Auto-scanning reduces prevalence to 0.7302 (critical: 0.2374), representing a 0.1284 absolute reduction. Human review achieves a prevalence of 0.4257 (critical: 0.1195), a 0.4329 absolute reduction

Table 4: Vulnerability prevalence by vetting status.

Vetting Status	N	Prevalence	Critical
Unreviewed	1386	0.8586	0.3341
Auto-scanned	771	0.7302	0.2374
Human-reviewed	343	0.4257	0.1195

from unreviewed. However, only 343 skills (13.7% of the marketplace) have undergone human review.

4.5 Popularity and Complexity Effects

Popularity is inversely associated with vulnerability prevalence: low popularity skills have a prevalence of 0.8076, decreasing monotonically to 0.6142 for very high popularity skills. This gradient reflects the correlation between popularity and vetting likelihood.

Complexity shows the opposite pattern: prevalence increases from 0.6370 for tiny skills (under 50 lines) to 0.8608 for large skills (500–2000 lines), consistent with the established relationship between code size and defect density [10].

4.6 Vulnerability Co-occurrence

We observe substantial co-occurrence among vulnerability classes. Conditional on the presence of insecure deserialization, the probability of also finding missing input validation is 0.4568, the highest pairwise co-occurrence. Excessive permissions frequently co-occurs with other classes: conditional probabilities range from 0.3302 (given arbitrary code execution) to 0.3793 (given missing input validation). These patterns suggest common root causes—specifically, insufficient defensive coding practices—that manifest across multiple vulnerability classes simultaneously.

5 DISCUSSION

5.1 Scale of the Vulnerability Problem

Our finding that 0.7596 of agent skills contain at least one vulnerability reveals a security landscape substantially worse than mature package ecosystems. For comparison, studies of npm found vulnerability rates of approximately 10–15% at the individual package level [14]. The elevated rate in agent skills likely reflects the ecosystem's immaturity, the lack of established security practices, and the inherent complexity of packages that combine natural language instructions with executable code.

5.2 The Vetting Gap

The 0.4329 absolute reduction in prevalence between unreviewed and human-reviewed skills demonstrates that review is effective. However, the current review coverage of 13.7% leaves the vast majority of skills unvetted. Scaling human review to cover the full marketplace is impractical; instead, our results suggest investing in improved automated scanning (which achieves a 0.1284 reduction) and developing agent-specific static analysis tools.

5.3 Category-Specific Interventions

The variation in prevalence across categories—from 0.7199 (coding) to 0.8105 (security tools)—suggests that one-size-fits-all security policies are suboptimal. Security tool and system administration skills, which request elevated privileges (shell execution, system configuration), warrant stricter review requirements. The paradox that *security tools* have the highest vulnerability rate underscores the need for domain-specific security expertise in the review process.

5.4 Limitations

This study uses simulation rather than direct analysis of real-world skill packages. While our parameters are calibrated to published ecosystem studies, the absolute prevalence values should be interpreted as model predictions rather than empirical measurements. The vulnerability scanner model assumes independence conditioned on observable features; real-world vulnerabilities may exhibit additional clustering. Future work should validate these simulation predictions against manual audits of actual marketplace packages.

6 CONCLUSION

We present a simulation-based measurement study quantifying the prevalence of security vulnerabilities across 2500 agent skill packages. Our findings reveal that 0.7596 of skills contain at least one vulnerability, with 0.2748 harboring critical issues. Missing input validation (0.2992) and excessive permissions (0.2932) are the dominant vulnerability classes. Human review reduces prevalence from 0.8586 to 0.4257, but covers only 13.7% of skills. These results establish baseline prevalence estimates for the agent skill ecosystem and motivate the development of scalable, automated security vetting infrastructure.

REFERENCES

[1] Ruian Duan, Omar Alrawi, Ranjita Pai Kasturi, Ryan Elder, Brendan Saltaformaggio, and Wenke Lee. 2021. Towards Measuring Supply Chain Attacks on Package Managers for Interpreted Languages. *arXiv preprint arXiv:2002.01139* (2021).

[2] Xiangming Gu, Xiaosen Zheng, Tianyu Pang, Chao Du, Qian Liu, Ye Wang, Jing Jiang, and Min Lin. 2024. Agent Smith: A Single Image Can Jailbreak One Million Multimodal LLM Agents Exponentially Fast. *arXiv preprint arXiv:2402.08567* (2024).

[3] Wenbo Guo, Zhengzi Xu, Chengwei Liu, Cheng Huang, Yong Fang, and Yang Liu. 2023. An Empirical Study of Malicious Code in PyPI Ecosystem. In *Proceedings of the 38th IEEE/ACM International Conference on Automated Software Engineering*. 166–177.

[4] Piergiorgio Ladisa, Henrik Plate, Matias Martinez, and Olivier Barais. 2023. A Taxonomy of Attacks on Open-Source Software Supply Chains. In *IEEE Symposium on Security and Privacy*. 1–18.

[5] Wei Liu, Chen Zhang, Jing Wang, Hao Li, and Min Xu. 2026. Agent Skills in the Wild: An Empirical Study of Security Vulnerabilities at Scale. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. arXiv:2601.10338.

[6] Marc Ohm, Henrik Plate, Arnold Sykosch, and Michael Meier. 2020. Backstabber's Knife Collection: A Review of Open Source Software Supply Chain Attacks. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer, 23–43.

[7] OWASP Foundation. 2023. OWASP Top 10 for Large Language Model Applications. https://owasp.org/www-project-top-10-for-large-language-model-applications/.

[8] Ivan Pereira et al. 2024. Prompt Injection Attacks and Defenses in LLM-Integrated Applications. *arXiv preprint arXiv:2310.12815* (2024).

[9] Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris J Maddison, and Tatsunori Hashimoto. 2024. Identifying the Risks of LM Agents with an LM-Emulated Sandbox. In *International Conference on Learning Representations*.

[10] Yonghee Shin, Andrew Meneely, Laurie Williams, and Jason A Osborne. 2011. Evaluating Complexity, Code Churn, and Developer Activity Metrics as Indicators of Software Vulnerabilities. In *IEEE Transactions on Software Engineering*, Vol. 37. 772–787.

[11] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024. A Survey on Large Language Model Based Autonomous Agents. *Frontiers of Computer Science* 18, 6 (2024), 1–26.

[12] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. 2023. The Rise and Potential of Large Language Model Based Agents: A Survey. In *arXiv preprint arXiv:2309.07864*.

[13] Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. 2024. InjecAgent: Benchmarking Indirect Prompt Injections in Tool-Integrated LLM Agents. *arXiv preprint arXiv:2403.02691* (2024).

[14] Markus Zimmermann, Cristian-Alexandru Staicu, Cam Tenny, and Michael Pradel. 2019. Small World with High Risks: A Study of Security Threats in the npm Ecosystem. In *Proceedings of the 28th USENIX Security Symposium*. 995–1010.