SADE: Safety-Aligned Dataset Engine

Sergio Valderrama
University of California, Santa Cruz
sevalder@ucsc.edu

Sallar Farokhi
University of California, Santa Cruz
sfarokhi@ucsc.edu

Abstract

Despite widespread use of flagship LLMs across the general public, ethical and privacy concerns persist, particularly regarding the generation of dangerous or sensitive information. While safeguards exist to prevent LLMs from violating safety guidelines, this remains an open challenge. Furthermore, such safeguards may inadvertently limit legitimate code generation use cases, especially for CLI commands involving privileged system operations.

This paper describes SADE [1], a scalable pipeline for generating potentially offensive command-line (CLI) actions, detailed within the Enterprise MITRE ATT&CK framework. This pipeline targets underrepresented platforms such as AWS, SalesForce, and PostgreSQL, and integrates structured technique representations, a curated platform knowledge base, graph-driven retrieval, and intention-first prompting. This pipeline aims to produce high-fidelity, auditable command datasets that support the evaluation and training of safety-critical AI systems, especially in agentic use-cases that involve enterprise software. We evaluate the datasets created using CMDCaliper [4], and assess the validity of the CLI code generated to suit the platform and MITRE technique. SADE establishes a baseline for future efforts in safety-conscious

1 Introduction

The rapid adoption of Large Language Models (LLMs) has evolved beyond simple code completion to power autonomous "Computer Using Agents" (CUAs) capable of executing complex multi-step operations [19]. While these models possess vast pretrained knowledge, their effectiveness in specialized security and enterprise contexts is often hindered by outdated knowledge bases and over-generalized safety alignment. A critical friction point arises in the domain of Command Line Interface (CLI) operations: a user or agent attempting to simulate a legitimate red-teaming attack path often faces "safety refusal," where the model incorrectly flags valid administrative commands as harmful [1]. Conversely, models may exhibit "hallucination," inventing non-existent syntax for proprietary cloud or SaaS tools. These limitations are particularly acute in rapidly evolving enterprise environments (e.g., AWS, Kubernetes, Azure) where documentation is fragmented, rendering standard training data insufficient for reliable agentic operations.

To address these reliability gaps, there is an urgent need for high-fidelity, structured datasets that map specific CLI actions to adversarial frameworks like MITRE ATT&CK. Such data is essential not only for improving the operational success of offensive security agents but also for hardening defensive systems. From a defensive standpoint, reliable labeled data allows for the training of "context-aware" detection models capable of distinguishing between benign administration and malicious "living-off-the-land" techniques—a "dual-use" dilemma that traditional signature-based detection fails to solve. Furthermore, for the safe deployment of CUAs, we require rigorous benchmarks to test alignment, ensuring agents can reject malicious instructions injected via indirect prompt attacks without refusing valid, high-stakes administrative tasks.

[1] `https://github.com/flacman/DatasetCreator` (Established Dec 10, 2025)

This paper presents SADE [2]: a Safety-Oriented Dataset Engine, designed to produce enterprise-specific, MITRE-classified command datasets that bridge these gaps. Unlike existing datasets such as CyPHER [4] and Raconteur [14], which primarily focus on OS-level shells with limited semantic coverage of modern cloud and SaaS platforms, our approach targets the "grey area" of enterprise software. By leveraging a combination of industry-grade cloud documentation and threat intelligence standards, we systematically generate realistic, platform-valid command examples. This pipeline enables the creation of knowledge bases that support both the training of robust detection algorithms and the development of capable, safety-aligned agents that can operate reliably within the Enterprise MITRE ATT&CK framework.

2 Related Work

Research surrounding CLI code generation and adversarial behavior using LLMs has grown substantially in recent years, but existing work for realistic, platform-diverse command generation remains in the early stages. A number of security-based corpora already provide partial coverage of adversarial CLI behavior, including Atomic Red Team [11], LOLBAS [7], Metta [16], RedCode [2], ThreatActorProcedures [13], and malware-oriented collections such as WhiteRabbitNeo [18], FireEye BUZZ [8], and theZoo [9]. Such datasets are modeled after ATT&CK methods and adversary playbooks, which aid in the procurement of CLI commands, but they're still subject to the cybersecurity concerns mentioned earlier. In addition, these datasets overwhelmingly target OS-native execution, while modern adversarial behavior increasingly spans cloud infrastructures, managed databases, and hybrid enterprise systems. CMDCaliper [4] is an embedding model that's designed to capture the semantic relationship across CLI commands. By applying CMDCaliper to the output of our data pipeline, we were able to assess the validity of the commands generated, and the semantic similarity to our knowledge bases. Our work extends the high-level goals of these datasets above but reorients them toward cross-platform, ATT&CK-grounded requests, enabling CLI generation that reflects platform-specific semantics, rather than isolated shell patterns.

On the other front, solutions for safeguarded LLMs have involved techniques such as vulnerability-to-technique mapping, malware behavior classification, and attack-graph construction. Each of these examples illustrates the role that the MITRE ATT&CK framework can play in describing adversarial intent, and by extension, prohibiting an LLM's response. By constructing structured technique-cards and embedding them in a typed knowledge graph, our approach treats ATT&CK as a generative blueprint for mapping adversarial intent. LLM safety research highlights the challenge of eliciting operationally sensitive content under modern safeguards. Studies on prompt injection, system-prompt extraction, and bypassing safety filters [6, 17, 5, 12] have shown that such filters are inconsistent and often triggered by various references to offensive intent, technique identifiers, or dangerous verbs. This pipeline could serve as a reproducible method to create safety-oriented datasets that allow LLMs to proactively access relevant, security-oriented CLI information.

3 Architecture

Our pipeline is designed to grant Large Language Models (LLMs) the ability to access security-conscious, platform-specific data at runtime, enabling the generation of informed CLI commands that operate outside the bounds of over-generalized safeguards. To achieve this, we engineer a pipeline capable of ingesting diverse documentation from varying platforms and frameworks, extracting a structured knowledge graph that facilitates safe, verifiable CLI code generation.

3.1 Knowledge Bases

The foundation of our architecture lies in two distinct Knowledge Bases (KBs), designed to mitigate the inherent unreliability of raw LLM generation in security-critical contexts. A primary motivation for constructing these specialized KBs is that the parametric knowledge of off-the-shelf models is neither sufficiently precise nor consistent for security-sensitive command generation.

Empirical studies on code generation consistently demonstrate that while LLMs can produce syntactically plausible code, they frequently exhibit non-syntactic bugs, API misuse, and inconsistent

[2] `https://github.com/flacman/DatasetCreator` (Established Dec 10, 2025)

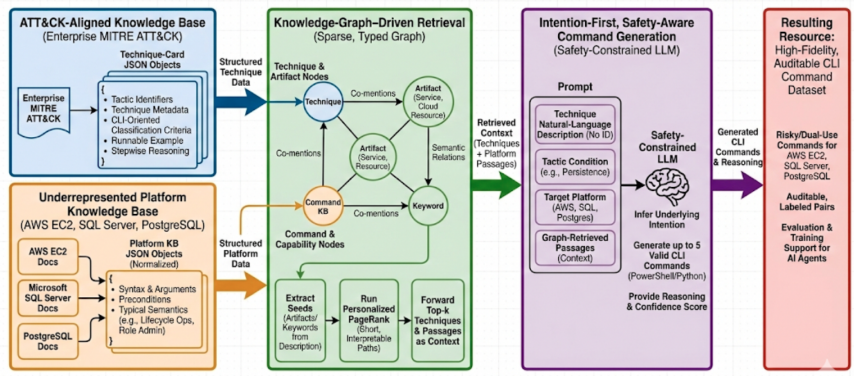

Figure 1: General Architecture of the Pipeline

adherence to specific framework versions [10]. This is particularly dangerous in the context of CLI operations, where naive prompting often results in the ”hallucination” of non-existent flags or the conflation of syntax between major version releases (e.g., mixing AWS CLI v1 and v2 parameters). Recent analyses of software supply chains highlight a similar systemic tendency for models to invent misleading dependencies or ”hallucinate” packages that do not exist [15], underscoring that plausible-looking output is not a proxy for correctness.

ATT&CK-Aligned Knowledge Base Furthermore, standard RLHF alignment tends to bias models toward a narrow subset of ”popular” administrative functions, leaving critical but niche security capabilities—such as specific IAM role configurations, snapshot export paths, or cross-account access mechanisms—effectively inaccessible. To address this, our approach generalizes the design philosophy of *Raconteur* [14]. While *Raconteur* utilizes document-augmented retrieval to explain obscure commands, our platform KB leverages a curated, versioned documentation graph to constrain the *generation* process. By grounding the LLM in a structured representation of valid platform operations, we ensure the model produces commands that strictly adhere to existing APIs while encouraging the exploration of the full breadth of the platform's security-relevant functionality.

Underrepresented Platform Knowledge Base A second repository captures platform-specific operational knowledge for AWS EC2, SQL Server, etc. While flagship LLMs are rarely denounced for antiquated information, rapidly changing infrastructure from popular software platforms can hinder the generation of platform-specific commands. In this pipeline, documentation from these platforms have been transformed into normalized JSON objects which encode syntax, operational semantics, one-shot examples and typical life-cycle operations such as database role administration. This knowledge base ensures that the system can generate valid commands across platforms that are weakly represented or prone to change.

3.2 Knowledge-Graph–Driven Retrieval

Both knowledge bases are fed into a sparsely typed knowledge graph that models the relationship between the techniques, command capabilities, and relevant keywords from the data. Implemented directly from HippoRag 2[3], the graph supports co-mention analysis between different techniques and platforms between platform resources and operational capabilities, and uses graph-based ranking to surface interpretable, contextually relevant relationships.

Given a natural-language query describing a tactic, platform, or operational (Technique) intent, the retrieval subsystem extracts seed nodes, performs personalized PageRank to identify the most relevant technique and platform passages, and forwards the top-k retrieved contexts. This ensures the generation module receives only targeted, high-relevance information, rather than full technique cards or raw documentation.

3.3 Intention-First, Safety-Aware Command Generation

A safety-constrained LLM processes the retrieved context along with the user-provided description of the tactic, condition, or platform. The model first infers the underlying operational intention (e.g., accessing a cloud resource or validating an account) and evaluates it within the system's safety constraints. If the intention is permissible, the LLM generates up to five syntactically valid, operationally consistent CLI commands (PowerShell or Python), accompanied by structured reasoning and confidence signals. The generation layer incorporates alignment rules that prohibit the emission of exploit chains or uncontrolled adversarial procedures. Instead, it produces commands that reflect platform behavior, administrative workflows, and life-cycle operations while remain auditable.

3.4 Resulting Dataset

The output of the generation pipeline is a structured, high-fidelity dataset [3] containing paired natural-language descriptions, safe reasoning traces, and valid CLI command examples for any platforms included in the knowledge base. Commands that carry dual-use risk are clearly annotated and traceable to their source reasoning, which makes it suitable for training platform-aware embedding models, evaluating agent safety, or constructing CLI commands aligned with the MITRE ATT&CK framework.

4 Evaluation/Results

To assess the performance of our pipeline, we generated embeddings for each of these knowledge bases using Nvidia's NV-Embed-v2 model. These datasets are then incorporated into an LLM with a naive implementation of HippoRAG [3], having the goal of generating CLI commands for platform-specific interfaces. We prompted Gemini 2.5 Flash to return the appropriate CLI commands that reflect each MITRE technique from our knowledge base, and as a result, we've extracted 12,339 commands from the cross section of these datasets.

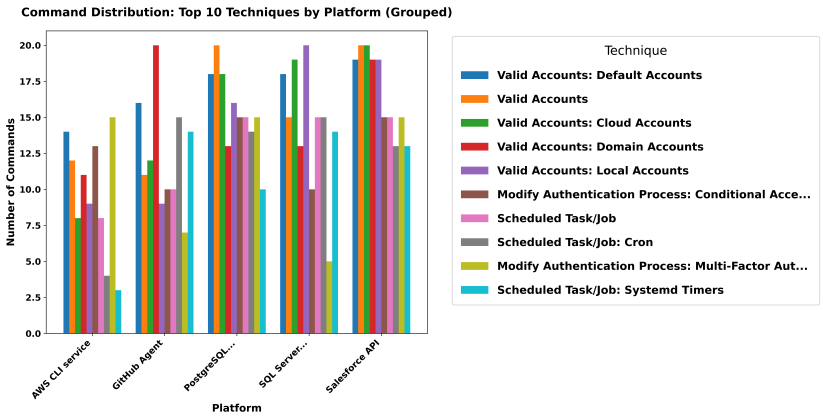

Figure 2: Most common techniques distributed across different platforms

Figure 2 shows the most prevalent MITRE techniques from the data pipeline, grouped by the platform. It's important to note that these were popular techniques by a very small margin, and every other sub-technique would have an amount between 5-10 commands. Despite the varying degrees of popularity across different platforms, The distribution of these techniques across platforms varies greatly. Services that don't involve as many client interactions have fewer mentions of performing MFA Authentication than platforms like AWS and Salesforce. In addition, despite the size of AWS's documentation over other services, there are considerably fewer commands targeting scheduled tasks or valid accounts. Observing the basic distribution of commands over the dataset can help identify commonly made patterns between the two knowledge bases.

[3] `https://drive.google.com/drive/folders/1DbmHzM7qkAc_kpK9ZcmY4hiX0634wMl1?usp=sharing`Established Dec 10, 2025

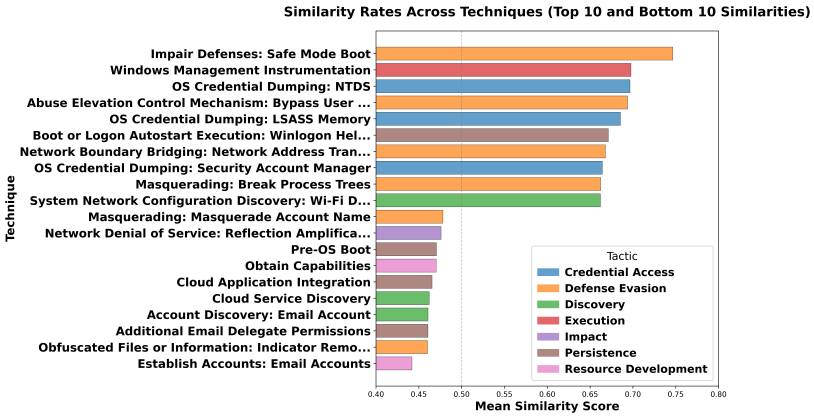

Figure 3: Techniques with varying similarities, across different tactics

To evaluate the fidelity of our generated CLI commands, we employed a dual-stage assessment strategy. First, we utilized CmdCaliper [4] as our embedding model, selected for its specialized pre-training on security-centric datasets, which allows it to capture the semantic nuances of administrative and malicious commands. Second, we deployed Llama 3.1 (8B) as an automated validator to verify command accuracy, specifically checking for syntactic validity and alignment with the target MITRE technique and platform. The validation results indicate a high success rate: the model confirmed that the vast majority of generated commands possess valid syntax and maintain strong semantic consistency with the source knowledge bases. Figure 3 shows the similarity scores between the CLI commands generated and the MITRE tactic it was associated with. Nearly all of the sub-techniques were bound with a range of 0.4-0.6, and such outliers are limited to rarely encountered cases within the platform knowledge base provided. These results substantiate our claim that the pipeline successfully achieves its primary objective of generating usable, high-quality CLI datasets, achieving thousands of class-balanced and varied records.

To assess the quality of our CLI commands, we employ CMDCaliper [4] as the embedding model, which is better suited to encapsulate the ATT&CK-specific shell output we've generated. A simple checker with Llama 3.1 is used to assess the accuracy of each CLI command, given the listed MITRE technique and platform. To our avail, the 8 billion parameter model claimed that nearly all commands generated by the data pipeline were considered to have valid syntax, and a strong semantic relation to the knowledge bases used. On this metric alone, we can claim that our pipeline has achieved the initial goal of generating usable correct CLI datasets. Figure 3 shows the similarity scores between the CLI commands generated and the MITRE tactic it was associated with. Nearly all of the sub-techniques were bound with a range of 0.4-0.6, and such outliers are limited to rarely encountered cases within the platform knowledge base provided. Across the techniques with the highest similarities, defense evasion and credential access are popular. This is to be expected, as these are examples of OS-specific MITRE techniques with established precedent.

5 Future Work

This work opens several avenues for further improvement across data realism, platform coverage, and safety evaluation. Expanding the knowledge base to additional enterprise environments—such as GCP, Azure, etc. would broaden the diversity of executable examples and better reflect modern hybrid architectures. Improving retrieval robustness and mitigating graph-propagated errors remain important challenges, as the fidelity of generated commands depends strongly on the quality and completeness of both the platform KB and the ATT&CK-aligned technique representations. Integrating dynamic execution validation or sandboxing could provide runtime correctness checks and reduce reliance on static verification alone.

Beyond data generation, our pipeline could support more advanced safety-research applications, including adaptive sampling for eliciting edge-case behaviors, multimodal (CLI + GUI) agent-behavior datasets, and stronger risk-assessment tools for model-assisted system operations. Finally,

leveraging these datasets to refine risk judges, evaluate LLM-based agents, and construct reproducible safety benchmarks offers a promising path toward more rigorous standards for secure model deployment.

6 Conclusion

We introduced a scalable, ATT&CK-grounded generation pipeline for producing realistic and platform-specific CLI commands across underrepresented cloud and database environments. By combining structured technique-cards, a curated platform knowledge base, a typed knowledge graph, and retrieval-guided intention-first prompting, the system generates high-fidelity and auditable operational examples while remaining compatible with modern LLM safety constraints. These datasets enable downstream applications such as platform-specific embedding training, red-team evaluation of AI agents, and reproducible benchmarking of safety-aligned command-generation systems. Ultimately, this work provides a foundation for more reliable, transparent, and secure model behavior in enterprise and agentic settings.

References

[1] Manish Bhatt et al. Cyberseceval 2: A wide-ranging cybersecurity evaluation suite for large language models. *arXiv preprint arXiv:2404.13161*, 2024.

[2] Chengquan Guo, Xun Liu, Chulin Xie, Andy Zhou, Yi Zeng, Zinan Lin, Dawn Song, and Bo Li. Redcode: Risky code execution and generation benchmark for code agents. In *NeurIPS 2024 Datasets and Benchmarks Track*, 2024.

[3] Bernal Jiménez Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. Hipporag: Neurobiologically inspired long-term memory for large language models. In *Advances in Neural Information Processing Systems 37*, 2024.

[4] Sian-Yao Huang, Cheng-Lin Yang, Che-Yu Lin, and Chun-Ying Huang. Cmdcaliper: A semantic-aware command-line embedding model and dataset for security research. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 20188–20206, 2024.

[5] Y. Li et al. R-judge: Benchmarking safety risk awareness for llm agents. *arXiv preprint arXiv:2401.10019*, 2024.

[6] P. Lin et al. Spe-llm: System prompt extraction in large language models. *arXiv preprint arXiv:2405.19320*, 2024.

[7] LOLBAS Project. Lolbas: Living off the land binaries, scripts, and libraries. Project Website, 2024.

[8] Mandiant / FireEye. Unauthorized access of fireeye red team tools and released countermeasures. CISA Advisory, 2020.

[9] Y. Tisf Nativ, Shahak Shalev, and Contributors. thezoo: Repository of live malware samples for research. GitHub Repository, 2014.

[10] Hammond Pearce, Baleegh Ahmad, Benjamin Tan, Brendan Dolan-Gavitt, and Ramesh Karri. Asleep at the keyboard? assessing the security of github copilot's code contributions. In *2022 IEEE Symposium on Security and Privacy*. IEEE, 2022.

[11] Red Canary Team. Atomic red team. Open-Source Adversary Simulation Library, 2024.

[12] M. Shu et al. Os-harm: A benchmark for measuring safety of computer use agents. *arXiv preprint arXiv:2506.14866*, 2025.

[13] Jacob Stickney. Threatactorprocedures–mitre att&ck collection. GitHub Repository, 2024.

[14] S. Subramani et al. Raconteur: A knowledgeable, insightful, and portable llm-powered shell command explainer. *arXiv preprint arXiv:2409.02074*, 2024.

[15] TechRadar Editorial Staff. Mitigating the risks of package hallucination and slopsquatting. TechRadar Online Article, 2024. Accessed: Dec. 2025.

[16] Uber-Common / Metta Project. Metta: Information security preparedness and adversary emulation tool. GitHub Repository, 2024.

[17] J. Wei et al. A taxonomy of jailbreaks and prompt injection attacks. *arXiv preprint arXiv:2402.06700*, 2024.

[18] WhiteRabbitNeo Organization. Whiterabbitneo cybersecurity models and resources. Hugging Face Model Hub, 2024.

[19] Zhexin Zhang et al. Agent-safetybench: Evaluating the safety of llm agents. *arXiv preprint arXiv:2412.14470*, 2024.