

Jailbreak Genome Scanner

Moses Mubvafhi

Roger Arendse

Leon Motaung

Mufaro Rukuni

With

Apart Research

Abstract

Jailbreak Genome Scanner (JGS) introduces a structural, pre-deployment safety pipeline that transforms jailbreak probing into a quantified systems-engineering process. The Jailbreak Arena Dashboard orchestrates multi-strategy attacker agents, real-time defender configuration (mock, API, or Lambda deployments), connectivity diagnostics, SSH-tunnel workflows, and streaming telemetry via threat-radar visualizations. New features include automatic Lambda Cloud instance discovery, vLLM endpoint validation, difficulty-tier prompt sampling, attacker readiness panels, and an integrated intelligence-gathering scraper.

Each prompt–response pair is classified by the Safety Classifier, embedded, and clustered into a “Genome Map,” and the Jailbreak Vulnerability Index (JVI) synthesizes exploit rate, mean severity, high-severity share, and failure-mode entropy into a 0–100 robustness score. Representative evaluations (10 attackers × 10 rounds) uncovered cross-domain vulnerability clusters, isolated critical-severity failures, and showed that attacker diversity combined with real-time intelligence ingestion sustains novel exploit discovery. The Continuous Defense module transforms detected patterns into prioritized vulnerabilities and adaptive guardrail rules, enabling a feedback loop from evaluation to mitigation. Overall, JGS provides a repeatable “Snyk-for-LLMs” workflow and a benchmarking primitive akin to “MLPerf for jailbreak robustness,” supporting actionable pre-release certification.

Keywords: multi-agent alignment, AI security, model evaluations, safety infrastructure.

1 Introduction

Modern alignment guardrails are routinely bypassed by prompt-level adversaries prior to deployment, with studies showing jailbreak success rates exceeding 30% in untested systems (Ganguli et al., 2022; Anthropic, 2023). The Jailbreak Genome Scanner (JGS) reframes jailbreak detection as a structural systems-engineering problem. Instead of testing isolated prompts, it composes adversarial prompt genomes, observes LLM failure phenotypes, and renders structural weaknesses on a “Genome Map” for pre-deployment sign-off.

This upgraded JGS introduces operational improvements including Lambda Cloud vLLM instance discovery, endpoint validation, firewall/SSH-tunnel diagnostics, difficulty-tier attacker sampling, structured prompt-database selection, real-time threat-intelligence ingestion, and mandatory BIO_HAZARD and CYBER_EXPLOIT attacker integration.

We investigate:

- **RQ1:** Do structural jailbreak maps provide deeper coverage than anecdotal prompts?
- **RQ2:** Does a competitive, adaptive attacker ecosystem produce more novel exploits?

We hypothesize:

- **H1:** Embedding-and-clustering reveal multiple stable cross-domain vulnerability clusters.
- **H2:** Attacker diversity + live threat ingestion sustains $\geq 25\%$ novel exploit discovery.

Summary of a sentence: JGS is a self-evolving stress test that pits attacker agents against target LLMs, maps structural weaknesses, and produces a standardized Jailbreak Vulnerability Index (JVI).

2 Methods

Structural Jailbreak Mapping

Adversarial prompts are sampled from the structured prompt database with difficulty tiers (L1–H10) and strategies including role-play, chain-of-command attacks, emotional coercion, translation inversion, cyber exploits, biohazard escalation, and policy probing. Defender responses are labeled (safe/unsafe, severity 0–5, domain), embedded, and clustered to form the Jailbreak Genome Map.

Jailbreak Arena Orchestration

The Streamlit dashboard configures defenders (Mock, OpenAI, Anthropic, Lambda). Lambda integrations support instance discovery, endpoint verification, firewall checks, and SSH-tunnel guidance. Operators adjust attacker count, rounds, and difficulty ranges and may enable a Lambda-based threat scraper to ingest novel jailbreak patterns from open repositories.

Defender Configuration: Supports Mock, OpenAI, Anthropic, and Lambda Cloud defenders. For Lambda deployments, the dashboard auto-detects active instances, pre-fills endpoints, displays IP metadata, validates vLLM availability, and guides users through firewall workarounds or SSH-tunnel setup.

Attack Configuration: Users select attacker count, rounds, and difficulty levels (Low, Medium, High, Custom). BIO_HAZARD and CYBER_EXPLOIT agents are forced into the pool to ensure safety-critical coverage.

Intelligence Gathering: A Lambda Web Scraper asynchronously collects real-time jailbreak patterns from GitHub and research forums, populating a panel of emerging threats.

Run-time Agent Initialization: A status panel displays readiness of BIO_HAZARD, CYBER_EXPLOIT, and other strategy generators.

Game-Based Live Evaluation

During the evaluation, attacker agents perform sequential rounds against the defender. Live telemetry includes exploit logs, radar animations, connectivity signals, and dynamic metric cards. Intelligence ingestion continues asynchronously, allowing newly discovered patterns to influence subsequent attacker sampling.

Quantitative Scoring

Plotly components compute:

- JVI gauge (0–100),
- exploit-rate time series,
- attacker-strategy distribution,
- severity histograms,
- attacker leaderboards.

The JVI integrates the exploit rate, the mean severity, the high-severity proportion, and the entropy of the failure modes to produce a reproducible risk rating.

Genome Visualization and Defense Feedback

A 3D explorer t-SNE visualizes clusters by domain, severity, or strategy. The Continuous Defense Improvement module analyzes patterns, surfaces high-impact vulnerabilities, and generates adaptive policy patches, completing the self-evolving evaluation → mitigation loop.

3 Results

A representative evaluation on a Lambda-hosted model (10 attackers × 10 rounds) produced the following:

- **JVI = 58/100**, indicating moderate risk.
- Exploit-rate spikes in rounds 7–8 driven by high-difficulty BIO_HAZARD commands.
- Chain-of-command and translation-inversion hybrids accounted for 45% of successful exploits.
- Three severity-5 failures were identified that required immediate defensive updates.
- Lambda-scraped attacker profiles ranked highest, demonstrating value of real-time intelligence.

- Genome Map analysis revealed cyber vs. bio clustering linked by a policy-probing ridge.
- Defense module recommended five prioritized vulnerabilities with auto-generated patches.

These results support both hypotheses: structural clustering (H1) and sustained exploit novelty (H2).

4 Discussion and Conclusion

The JGS pipeline elevates LLM safety evaluation from anecdotal probing to a structural, quantitative discipline. The Arena Dashboard unifies defender configuration, adversarial testing, intelligence ingestion, telemetry, scoring, and automated defense feedback.

Structural vulnerability mapping validates RQ1, while attacker diversity and intelligence ingestion validate RQ2. Challenges ahead include scaling embeddings, extending the classifier to multimodal content, and formalizing regulatory JVI thresholds.

JGS provides foundational infrastructure for Snyk-style continuous monitoring and MLPerf-style standardized jailbreak benchmarking. Future work includes attacker LLMs, reinforcement-learning red teams, and integrating defense rules directly into model-serving pipelines.

References

- Anthropic. (2023). *Constitutional AI: Harmlessness from AI Feedback*. arXiv:2306.
- Ganguli, D., Lovitt, L., Kernion, J., Askell, A., Bai, Y., Gopal, S., et al. (2022). *Red Teaming Language Models to Reduce Harms*. arXiv:2209.07858.
- OpenAI. (2023). *GPT-4 Technical Report*. arXiv:2303.08774.
- Perez, E., Ringer, S., Kramár, J., et al. (2022). *Discovering Language Model Behaviors with Model-Written Evaluations*. arXiv:2212.09251.

Appendix — Security Considerations

All adversarial prompts, scraped intelligence, and LLM responses remain confined to secure sandboxes. Harmful outputs are blocked from display and redacted by the Safety Classifier. Threat scrapers and attacker tools operate under isolation, with encrypted storage and audit logging. Future improvements include automated PII scrubbing, tamper-evident audit logs, and formal data-retention policies for regulatory review.