

OPENAI CYBERSECURITY GRANT PROGRAM PROPOSAL

---

# **agent-audit-kit: Defensive Evaluation Toolkit for Tool-Using AI Agents**

*An open-source toolkit for evaluating agent permissions, auditability, prompt/tool injection paths,  
and risky autonomous actions.*

---

**Sattyam Jain**

GenAI Tech Lead, Attri.ai

sattyamjain96@gmail.com

<https://www.linkedin.com/in/sattyamjain/>

<https://www.sattyamjjain.in/>

<https://github.com/sattyamjjain>

Date generated: 28 July 2026

## 1. Executive Summary

---

agent-audit-kit is an early-stage open-source defensive security toolkit for evaluating tool-using AI agents. The project focuses on practical failure modes in agent systems: excessive tool permissions, weak audit logs, unsafe integrations, prompt/tool injection paths, missing human checkpoints, and risky autonomous actions.

The aim is to help open-source maintainers answer simple but important questions before connecting agents to real systems: what tools can this agent call, what data can it touch, what actions can it take, what evidence is logged, and where should human approval be required?

This is not an offensive-security project. It will use synthetic examples and toy tools to make agent security failures visible in a safe, repeatable way.

## 2. Problem

---

AI agents are starting to call tools, read repositories, access SaaS systems, run code, and take actions across real workflows. The security problem is that many agent systems still treat tool access as a product feature, not as a permissioned security boundary.

Today, it is hard for maintainers to answer basic questions:

- What tools can this agent call?
- What data can it touch?
- What action was taken?
- Why was the action taken?
- Can access be revoked?
- Did a prompt or tool response influence the agent into doing something unsafe?
- Is there enough audit evidence to debug or review the behavior later?

This project focuses on that gap. The goal is to make safer defaults easier for open-source builders by providing practical checks, synthetic tests, and example patterns.

## 3. Proposed Work

---

agent-audit-kit will have three main outputs.

### A. Agent permission and audit checklist

A concrete checklist for agent builders and maintainers. It should cover:

- tool inventory
- permission scopes
- revocation behavior
- human approval points
- audit events

- sensitive action boundaries
- unsafe default patterns
- prompt/tool injection paths
- logging and observability gaps
- CI/security review readiness

The checklist should be written for working engineers, not only security specialists.

## **B. Evaluation harness for tool-using agents**

A lightweight Python-based evaluation harness that can run synthetic defensive tests against agent workflows. The harness should not require real secrets or real customer systems. It should use toy tools and synthetic data to check whether an agent:

- calls tools outside the intended scope
- acts on untrusted tool output without validation
- exposes sensitive-looking synthetic data
- skips approval for risky actions
- produces insufficient audit logs
- fails to explain or trace tool decisions
- handles malicious or confusing tool responses poorly

The goal is to make these failure modes visible before an agent is connected to real systems.

## **C. Example agent integrations and safer patterns**

Small example agents showing unsafe and safer designs. These examples should demonstrate:

- scoped tool access
- explicit approval gates
- structured audit logs
- deny-by-default behavior
- safe handling of untrusted tool responses
- CI-friendly checks

Where useful, prior work on scoped and revocable permissions (agent-airlock) may serve as one example integration pattern. agent-audit-kit remains the focus of this proposal.

# **4. Methodology**

---

The methodology is deliberately simple and repeatable:

- Define the threat model and scope.
- Build synthetic tools and synthetic datasets.
- Create repeatable evaluation cases.

- Run evaluations across selected OpenAI models and agent configurations.
- Compare unsafe and safer implementations.
- Publish findings, limitations, and recommended safe defaults.

The project may use OpenAI models through the API for evaluation scenarios and agent behavior tests. It may test across current frontier and lower-cost models where useful, but it should not depend on one model only. The focus is the security behavior of the agent system around the model, not the model in isolation.

## 5. Datasets and Data Handling

---

No private data will be used. No customer data, confidential data, or personal data will be included.

Any dataset created will be synthetic and published with the project. It may include:

- toy prompts
- synthetic tool responses
- synthetic secrets
- synthetic enterprise records
- expected pass/fail outcomes for agent behavior
- audit-log examples

The dataset will be designed only for defensive evaluation.

## 6. Public Outputs

---

Expected public outputs:

- open-source code under an OSI-approved license
- synthetic evaluation datasets
- documentation and threat model
- example integrations and runbooks
- CI-friendly usage examples
- a short final report with results, limits, and open problems

Results will be shared publicly for maximal public benefit.

## 7. Timeline

---

### September 2026

Finalize scope, license, project structure, threat model, and defensive evaluation plan.

## October 2026

Build the first version of the synthetic tool-use evaluation harness, including toy tools, test cases, expected outcomes, and audit-log checks.

## November 2026

Add example unsafe and safer agent integrations. Document permission patterns, revocation behavior, human checkpoints, and CI usage.

## December 2026

Run evaluations across selected OpenAI models and agent configurations. Publish findings, limitations, and recommended safe defaults.

## January 2027

Package the first stable release. Publish documentation, synthetic dataset, examples, and final technical report.

## 8. Budget and Resources

---

Requested support: USD 25,000 plus API credits.

Use of funds:

- engineering time to build and document the toolkit
- synthetic dataset and evaluation-case creation
- testing across multiple model configurations
- CI, release, documentation, and security hygiene
- public report and example integrations

API credits would be used for running repeatable evaluations across agent scenarios and model configurations.

If only API credits are possible: a smaller version can still be built around the evaluation harness and synthetic test cases.

If partial funding is possible: USD 10,000 plus API credits would support a narrower MVP — threat model, a basic evaluation harness, 20-30 synthetic test cases, and initial documentation.

With full requested support: the target is a complete public release with examples, CI-friendly checks, documentation, and a written technical report.

## 9. Applicant Fit

---

Sattyam Jain is a GenAI Tech Lead at Attri.ai. His day-to-day work is production AI engineering: agent workflows, model routing, tool execution, memory, evaluations, observability, audit logs, access controls, and customer-facing deployments.

Outside work, he has been building early open-source projects around agent security:

- agent-audit-kit, the primary project for this proposal, focused on security checks and audit patterns for agentic systems
- agent-airlock, related supporting work focused on scoped and revocable agent permissions, and a possible example integration for agent-audit-kit's safer-pattern examples
- Provael, related work around red-team and evaluation evidence for robot / VLA policy behavior in simulation

These projects are early-stage open-source work, not widely adopted infrastructure yet. That is precisely why a small, focused grant would help. The funding would help turn agent-audit-kit into a cleaner, documented, tested toolkit that other maintainers can run and critique.

## 10. Safety and Responsible Disclosure Boundary

---

This is a defensive project. It will not build offensive agents, malware workflows, exploit automation, or tools for attacking real systems.

Any adversarial cases will be synthetic and limited to demonstrating defensive checks. The project will not publish unpatched vulnerabilities in third-party systems. If a real vulnerability is discovered during related work, it will be handled through responsible disclosure.

## 11. Contact

---

Sattyam Jain

- Email: [sattyamjain96@gmail.com](mailto:sattyamjain96@gmail.com)
- LinkedIn: <https://www.linkedin.com/in/sattyamjain/>
- GitHub: <https://github.com/sattyamjjain>
- Website: <https://www.sattyamjjain.in/>