Global Digital Forensics · Guide edition 2026-11-03 · evestigate.com/insights/ai-red-teaming-with-reproducible-findings/

AI SECURITY

# Reproducible AI Red Teaming for Prompt Injection and Leakage

An AI assistant wired to your documents, mailboxes and APIs can be turned against you by a single crafted input, and the findings your engineers can reproduce are the ones they can fix and defend. Test the whole pipeline with a method that records every variable. Call (800) 868-8189 at any hour.

Published September 3, 2026 · Updated October 30, 2026

By Robert Knudsen, Computer forensics and cybersecurity examiner, Global Digital Forensics

## Moving AI Security Testing Beyond Novelty Jailbreak Demos

As corporations rapidly integrate large language models, retrieval augmented generation pipelines and autonomous AI agents into customer portals, legal workflows and internal knowledge bases, they open an entirely new class of application security risk. However, many early AI red team assessments amount to little more than novelty demonstrations where a tester tricks a chatbot into writing a rude poem, which gives engineers little to act on. Real enterprise AI risk occurs when an attacker uses indirect prompt injection hidden inside an incoming email, resume or document to hijack an AI assistant, extract confidential SharePoint records across permission boundaries or trigger unauthorized downstream API actions.

## How We Test AI Systems With Reproducible Engineering Rigor

Our application penetration testing and AI security architects evaluate enterprise AI deployments using the same disciplined, reproducible methodology we bring to critical infrastructure and financial software. We test the complete AI system architecture: model input and output guardrails, vector database retrieval permissions, system prompt confidentiality, plugin and function calling authorization controls and training data poisoning paths. For every vulnerability uncovered, we document the exact deterministic or statistical reproduction steps, the underlying architectural root cause, the concrete business data exposed and a verified engineering remediation blueprint followed by retesting.

## What a Reproducible Finding Must Contain

A finding against a language model system is only useful if a second engineer can make it happen again, so we record every variable that shaped the result. That means the exact input sequence, the system prompt version in effect, the model identifier and provider, the sampling parameters, the retrieval corpus state and the documents returned, the tool schema exposed to the model, the user identity and permission set under which the session ran and the gateway or application logs that captured the exchange. Where the model behaves probabilistically we report the outcome across repeated trials rather than a single success, and we preserve the raw transcripts so the rate can be checked. A finding written this way survives the engineering team's first question, which is always whether it is real.

© 2026 Global Digital Forensics, Inc. All rights reserved.

The report also records the conditions each finding depends on. A behavior demonstrated against one model version is tied to that version and retested after the provider updates weights or safety tuning, and a statistical finding is reported as an observed rate and range across a stated number of trials. Guardrail products from third parties are evaluated by their observable behavior, which is how we test any control whose rules the vendor keeps private. When a retrieval permission failure depends on a specific index state, the finding is tied to that state and we record the conditions under which it recurs. Recording those conditions protects the client twice: the engineers fix what is actually broken, and if the assessment is ever examined by a regulator, an insurer or a court, every claim in the report rests on the evidence behind it. Our AI security consulting work is built on that discipline.

## The Frameworks That Define the Scope

Three public references shape how we scope a test. The OWASP Top 10 for Large Language Model Applications names the failure classes the engineering community has agreed matter most, among them prompt injection, sensitive information disclosure, improper output handling, excessive agency, system prompt leakage and weaknesses in vector and embedding stores. Each of those maps to a test family in our methodology, and the report cross references them so a security lead can show coverage against a recognized list. The OWASP API Security project governs the other half of an agentic system, the function calls and downstream APIs the model is permitted to invoke, where broken object level authorization and excessive data exposure appear in a new costume.

The NIST AI Risk Management Framework organizes the governance side around four functions, Govern, Map, Measure and Manage, and its generative AI profile, already cited in this guide, lists the risks specific to generative systems. We use the framework to connect test results to the risk register the organization already keeps, so a finding about data leakage through retrieval is recorded as a measured risk with an owner rather than as an isolated bug. For a regulated deployment, that mapping is what lets the compliance team show that the organization identified, measured and managed the risk, which is the question regulators and plaintiffs ask after an incident.

## Phases of an AI Assessment That Engineers Can Act On

The engagement begins with a threat model and a data flow map, because an AI system's attack surface is the set of places untrusted content can enter and the set of actions the model can cause. We identify every ingestion path, from user chat to uploaded files, retrieved documents, email, web content and tool results, and every effector, from API calls and database writes to messages sent on a user's behalf. We also inventory what data the retrieval layer can reach and under whose permissions. Scoping with that map means the test covers the paths an attacker would use rather than the chat window alone, and it tells the engineering team in advance which systems will see test traffic.

Testing then works outward from the model boundary. Direct and indirect prompt injection is attempted through each ingestion path, with payloads embedded in documents, calendar invites, web pages and tool outputs to see whether the model follows instructions it should treat as data. Retrieval is tested for permission boundaries by asking as one user for content another user owns. Tool and agent authorization is tested by inducing the model to call functions with parameters outside the user's entitlement. Output handling is tested for script and query injection where model output reaches a browser or a database, and exfiltration channels

© 2026 Global Digital Forensics, Inc. All rights reserved.

such as rendered images and links are checked. Each confirmed finding is documented with its reproduction record, its root cause in the architecture and a remediation the engineers can verify, and a retest closes the loop. Our application penetration testing practice supplies the methodology these tests extend.

The report is written for two readers at once. Engineers receive each finding with its reproduction record, the architectural root cause, the specific remediation and the retest result, so the fix starts from the record itself rather than from the tester's notes. Security leadership receives a narrative that explains what the system can be made to do, under what conditions and with what business consequence, mapped to the OWASP and NIST categories so coverage can be shown to a regulator, a customer or a board. Where a finding has a statistical character, the narrative states the observed rate and the number of trials rather than rounding it to a yes or a no. Both readers see the same facts in the form each needs, which is what keeps the remediation and the governance record aligned.

## How Teams Prepare Before and After the Test

Strong programs treat the provider's safety filter as one layer among several. Those filters address harmful content, while your authorization model governs which documents a model may retrieve for a given user, so the authorization model is tested directly. The test covers the chat interface and also the documents, emails and web pages the model reads on its own, which is where much of the real exposure sits. Each of the assistant's tools runs under a service account scoped to its task, so a successful injection reaches only what that tool needs. Prompts, retrieved context and tool calls are logged from the first day of deployment, which gives a later cloud forensics engagement a forensic record to work from.

Preparing for the assessment puts those practices in place before testing begins. The security lead gathers the architecture diagram, the model and provider inventory, the system prompts in use, the tool definitions and the data classification of the retrieval corpus. The platform team enables and preserves gateway logs so the test traffic itself is recorded. Product owners identify the actions the system is permitted to take and the actions that stay outside its authority, and both lists become test cases. After the test, the report becomes the baseline for the next release. Models and prompts change often, and each release is a chance to confirm that closed findings stay closed, which is why many clients move from a point in time assessment to continuous AI testing with human review of the results.

Reading the results as a CISO starts with what the system was shown to do. A finding that the assistant followed instructions embedded in a document and sent a summary to an outside address is a demonstrated capability with a business consequence, and it belongs in the risk register with an owner and a date. A finding that a guardrail could be bypassed to produce an off topic answer is a quality issue, and it moves into the risk register when the bypass also reaches data or actions. The legal team reads for the first category when an incident has already occurred, because the question will be whether the organization knew the behavior was possible and what it did about it. The report's mapping to the NIST functions is built to answer that question with dates and evidence.

© 2026 Global Digital Forensics, Inc. All rights reserved.

## Secure Your AI Deployment Before Production Rollout

Call Global Digital Forensics at (800) 868-8189 to scope a reproducible AI red teaming and application security assessment with our senior engineers. The initial consultation is free, and our AI security consulting page explains how a test is scoped around the systems, data and actions your deployment can reach.

## What a Reproducible AI Finding Must Record

A finding against a language model system earns its value when a second engineer can reproduce it. This table lists the variables a finding records, why each one matters and how the examiner reads it, so a security lead can judge whether a report will survive the engineering team's first question. The last column explains why a statistical finding is reported as a rate across repeated trials, and why a behavior shown against one model version is retested after the provider updates weights.

Variables captured for a reproducible AI finding, the reason for each and how the examiner reads it.

| Variable recorded | What it protects | How the examiner reads it |
|---|---|---|
| **Exact input sequence** | Lets a second engineer repeat the attempt | Rerun after each model update |
| **System prompt version** | Fixes the instruction context in effect | Recorded for each release, since prompts change often |
| **Model identifier and provider** | Ties the result to a specific model | Retested when the provider changes weights |
| **Sampling parameters** | Explains probabilistic variation | Reported across repeated runs |
| **Retrieval corpus state and documents returned** | Shows what the model could reach | Tied to the index state at test time |
| **User identity and permission set** | Establishes the entitlement under test | Repeated for each role in scope |
| **Gateway and application logs** | Captures the exchange independently | Preserved wherever logging is enabled |

© 2026 Global Digital Forensics, Inc. All rights reserved.

## How to Begin and What Happens on the First Call

Have ready an architecture diagram or a verbal description of the system, the models and providers in use, the data sources the retrieval layer can reach, the tools or APIs the model can call, the user roles the system serves and the date of the next release or rollout. Our engineer asks where untrusted content enters, what the system is permitted and forbidden to do, whether a test environment with representative data exists, what logging is in place and whether compliance or outside counsel will rely on the report. You then receive a written scope that names the ingestion paths, tools and data in bounds, a timing plan keyed to your release schedule and a proposal for the assessment, the reproduction records and the retest. The initial consultation is free, and the hotline at (800) 868-8189 is answered 24/7.

## Technical References and Further Reading

Primary sources and provider documentation behind this guide: NIST Generative AI Profile.

Authorities cited in this guide: OWASP Top 10 for Large Language Model Applications. Names the failure classes, including prompt injection, excessive agency and system prompt leakage, that organize the test families in this guide, in the project's current published list. NIST AI Risk Management Framework. Supplies the Govern, Map, Measure and Manage structure used to connect findings to the organization's risk register, in the current NIST publication. OWASP API Security Project. Supplies the authorization and data exposure categories applied to the function calls and downstream APIs an agentic system can invoke, in the project's current published list. NIST Generative AI Profile. Lists the risks specific to generative systems that the assessment measures, in the published companion profile to the AI Risk Management Framework.

FREQUENTLY ASKED

## Jailbreaks, Timelines, Cost and Production Safety

### What is the difference between AI red teaming and a jailbreak test?

A jailbreak test asks whether the model can be made to say something its content policy forbids. AI red teaming in an enterprise setting asks whether the system around the model can be made to do something your authorization model forbids: retrieve another user's records, call an API with parameters the user is not entitled to, follow instructions hidden in a document it was asked to summarize or leak its system prompt and the data in it. The second set of questions is where financial and legal exposure lives, and it is answered by testing the whole pipeline around the model.

### How long does an AI security assessment take?

The drivers are the number of ingestion paths and tools the system exposes, the size and sensitivity of the retrieval corpus, how many user roles and permission sets must be tested, whether the engineering team can provide a test environment with representative data and how quickly findings are remediated for retest. A single assistant over one document store scopes smaller than an agent with a dozen tools across several business systems. We set the schedule during scoping and deliver confirmed findings as they occur rather than holding them for the final report.

© 2026 Global Digital Forensics, Inc. All rights reserved.

### What determines the cost of AI red teaming?

The initial consultation is free. The proposal reflects the architecture's complexity, the number of models, prompts, tools and data sources in scope, the depth of statistical testing required for probabilistic behaviors, whether source code and infrastructure configuration review is included and the reporting that compliance or the board will need. A continuous testing arrangement spreads the cost across releases and is often preferred for systems that change frequently.

### Will the test expose our data or disrupt production?

Testing is scoped to a controlled environment whenever one exists, with representative data and tool endpoints that simulate real actions. Where production testing is necessary, rules of engagement define the accounts, the data and the actions in bounds, and any exfiltration tests use marker data we plant rather than live records. All transcripts and findings are handled under the terms of the engagement agreement and, where counsel directs the work, under that direction. We also tell you which system components will see test traffic so your monitoring team knows what to expect.

### Can an AI red team report be used with regulators or in litigation?

A report built on reproducible findings, documented methodology and stated conditions is the kind of record that supports a position before a regulator or in a dispute, because another expert can check it. We write with that audience in mind and we have supported attorneys with technical declarations and testimony in federal and state courts in other areas of our practice. Your lawyers decide how a particular report is used, and bringing them into scoping early keeps that option open.

## Web Addresses for the Sources Cited

1. OWASP Top 10 for Large Language Model Applications: https://owasp.org/www-project-top-10-for-large-language-model-applications/
2. OWASP API Security project: https://owasp.org/API-Security/
3. NIST AI Risk Management Framework: https://www.nist.gov/itl/ai-risk-management-framework
4. NIST Generative AI Profile: https://www.nist.gov/publications/artificial-intelligence-risk-management-framework-generative-artificial-intelligence

Reviewed by Joseph Caruso, Founder, CEO and CTO, CISSP. Last reviewed November 3, 2026.

## Start With a Free Consultation

The initial consultation is free, and requests are handled in confidence. On the first call a senior member of our team listens to the question you need answered, identifies the records and systems involved, and recommends the next step.

**Call (800) 868-8189**, answered 24/7, request a consultation at evestigate.com/contact, or email info@evestigate.com.

© 2026 Global Digital Forensics, Inc. All rights reserved.