ODA3 INSTITUTE

Practitioner Cheat Sheet

AI Red Teaming Methodology

DocID: ODA3-2026-07-CHT-SEC-003
Classification: Public

1. Executive Overview

AI red teaming is structured adversarial testing of an AI system to find failures — security, safety, and behavioral — before an adversary or a real-world deployment does. It differs from both traditional penetration testing and standard model evaluation. Penetration testing targets deterministic infrastructure with known-bad inputs; model evaluation measures average-case quality against a benchmark. AI red teaming targets the *worst case* of a probabilistic system: the crafted input, the multi-step interaction, or the environmental condition under which the model does something it should not [T2].

This is a methodology reference, not an attack cookbook. It covers how to scope, authorize, staff, execute, and report an AI red team engagement, and how to convert findings into control decisions — deliberately without publishing working exploit payloads, jailbreak strings, or step-by-step attack recipes, which belong inside an authorized engagement, not a public document. The distinction matters for governance: a red team program's value is in its process discipline and its evidence trail, not in any single clever prompt.

> **Key Takeaway**
>
> Red teaming is a repeatable assurance process, not a one-time hacking event. Its output is not "we broke it" — it is prioritized, reproducible findings mapped to controls, with enough rigor that a second team could reproduce the result and an assessor could trust the report. Authorization, scope, and safety handling come *before* any testing begins.

Methodology Note

This document synthesizes publicly available evidence from standards organizations, open specification projects, regulatory publications, academic research, and vendor security advisories. No ODA3 proprietary datasets, client engagement data, or internal red team results are referenced, consistent with ODA3 Institute's founding in March 2026. Evidence is tiered: T1 = Primary Verified (standards, regulatory documents, vendor advisories, documented incidents); T2 = Secondary Verified (credible reporting corroborated by ≥ 1 independent source); T3 = Controlled Simulation / Academic Proxy (lab-replicated or peer-reviewed); T4 = Anecdotal / Unverified. Where sources disagree, differing viewpoints are identified rather than resolved into an unsupported conclusion.

2. Key Concepts & Glossary

Term	Definition
AI Red Teaming	Structured adversarial testing of an AI system to surface security, safety, and behavioral failures before deployment or exploitation
Penetration Testing (contrast)	Authorized testing of infrastructure/application security against known vulnerability classes; typically deterministic, often third-party for audit
Model Evaluation (contrast)	Average-case measurement of model quality/safety against a benchmark or golden dataset; not adversarial by design
Rules of Engagement (RoE)	The authorized scope, targets, methods, timing, and constraints agreed before testing begins; the document that makes testing lawful and safe
Attack Objective	What the red team is trying to make the system do (e.g., leak data, take an unauthorized action) — distinct from the technique used to achieve it
Jailbreak	An input crafted to bypass a model's safety alignment or refusal behavior
Prompt Injection (direct/indirect)	Adversarial instructions supplied directly by the user, or embedded in content the system later retrieves
Harm Category	The class of failure being tested (e.g., CBRN uplift, data exfiltration, harmful content, bias) — NIST AI 600-1 defines twelve for generative AI

Term	Definition
Coverage	The proportion of in-scope harm categories, attack surfaces, and lifecycle stages actually exercised by an engagement
Reproducibility	The property that a documented finding can be re-triggered by another tester following the report — the difference between a finding and an anecdote
Attack Success Rate (ASR)	The fraction of attempts that achieve the attack objective; meaningful only with a defined objective, sample size, and target version
Automated Red Teaming	Tooling that generates and runs adversarial inputs at scale (fuzzing, evolutionary prompt search, judge-scored campaigns)
Manual/Expert Red Teaming	Human-driven testing applying domain expertise, creativity, and multi-step reasoning that automation misses
Human/AI Red Teaming	NIST AI 600-1 approach combining human testers with AI assistance to scale and diversify adversarial coverage
Judge Model	A secondary model that scores whether an attack attempt succeeded; itself a source of measurement error if poorly calibrated
Blast Radius	The maximum damage achievable once an attack objective is met, determining finding severity independent of exploit difficulty
Finding Severity	A risk rating combining impact (blast radius) and exploitability (ease, prerequisites, reliability)
Remediation Retest	Re-execution of a finding's attack path after a fix, to confirm closure rather than mere suppression
Disclosure Handling	The controlled process for documenting, storing, and communicating findings — especially sensitive ones — without creating new exposure

3. Technical Deep Dive

3.1 Red Teaming vs. Adjacent Practices

Dimension	Penetration Testing	Model Evaluation	AI Red Teaming
Target	Infrastructure, application	Model average-case quality	System worst-case behavior
Input style	Known vulnerability classes	Representative benchmark	Adversarial, crafted, multi-step
Determinism	Largely deterministic	Statistical average	Probabilistic worst-case
Primary question	Is it exploitable?	Is it good enough?	What can it be made to do?
Typical output	Vulnerability list	Score/benchmark result	Reproducible findings + severity + control mapping

The three are complementary, not substitutes. A mature AI assurance program runs all three; this document addresses the third.

3.2 The Four Assessment Areas

Verified industry guidance (OWASP GenAI Red Teaming Guide) frames AI red teaming across four areas, and a complete engagement should consciously decide its coverage of each [T1]:

- **Model evaluation** — testing the base model itself: alignment robustness, refusal reliability, bias, toxicity, and susceptibility to jailbreaks independent of the surrounding application
- **Implementation testing** — testing the application wrapped around the model: prompt construction, RAG pipeline, guardrails, input/output handling, and API surface
- **Infrastructure assessment** — testing the hosting, access controls, secrets handling, tool/plugin integrations, and the agent execution environment

- **Runtime behavior analysis** — testing the system live: multi-turn interactions, agentic tool-use loops, memory effects, and behavior under sustained or adversarial load

Skipping an area is a legitimate scoping choice; skipping it *silently* is a coverage gap that will surface in assessment.

3.3 The Engagement Lifecycle

A defensible engagement runs a fixed sequence. The order is not optional — authorization and safety handling precede any testing.

```
1. Authorization & Rules of Engagement
   (scope, targets, methods, timing, legal sign-off,
    data-handling and safety constraints)
        ▼
2. Threat Modeling & Objective Setting
   (which harm categories, which attack surfaces,
    which lifecycle stages — mapped to NIST AI 600-1
    risk categories and OWASP LLM/Agentic Top 10)
        ▼
3. Test Planning
   (manual vs. automated split, tooling, sample sizes,
    judge/scoring method, success criteria per objective)
        ▼
4. Execution
   (run campaigns; log every attempt, input, output,
    model/version, and outcome for reproducibility)
        ▼
5. Triage & Severity Scoring
   (dedupe, confirm reproducibility, score impact ×
    exploitability, discard non-reproducible anecdotes)
        ▼
6. Reporting & Control Mapping
   (prioritized findings → recommended controls →
    owners; sensitive findings via disclosure handling)
        ▼
7. Remediation Retest
   (re-run the attack path after fixes; confirm
    closure, not just suppression of one variant)
```

3.4 Threat Modeling: Anchoring to Verified Frameworks

Objective-setting should map to established risk catalogs rather than an ad-hoc list, so coverage is defensible and findings are portable:

- **NIST AI 600-1** defines twelve generative-AI risk categories (including CBRN information/capabilities, confabulation, data privacy, information security, information integrity, harmful bias, and value-chain/component integration) and four red-teaming approaches — General Public, Expert, Combination, and Human/AI [T1]. A red team plan should state which of the twelve are in and out of scope, and why.
- **OWASP Top 10 for LLM Applications** and the **OWASP Top 10 for Agentic Applications (2026)** provide technique- and risk-level anchors; findings mapped to these IDs are immediately intelligible to other teams [T1/T2].
- **MITRE ATLAS** provides an adversarial technique taxonomy for AI systems; useful for structuring *how* an objective might be pursued, provided technique IDs are verified against the current published matrix rather than cited from memory [T2].

3.5 Manual vs. Automated Testing

Aspect	Automated Red Teaming	Manual / Expert Red Teaming
Strength	Scale, breadth, regression, repeatability	Creativity, multi-step reasoning, novel objectives, context
Weakness	Misses novel/contextual attacks; judge-model error	Does not scale; harder to reproduce without discipline
Best for	Known attack-class coverage, pre-release gates, drift regression	New systems, high-stakes targets, agentic multi-step abuse

The two are complementary. NIST's Human/AI approach formalizes combining them; mature programs use automation for breadth and coverage regression, and expert testers for depth and novelty [T1]. Open-source frameworks commonly used for the automated dimension include Microsoft PyRIT, Garak, and Promptfoo, among others; tooling choice matters less than disciplined coverage, versioned test sets, and calibrated scoring [T2]. A judge model used to score automated campaigns is itself a measurement instrument — its calibration and false-positive/negative rates should be documented, since an over-lenient judge inflates apparent success and an over-strict one hides real findings.

3.6 Agentic and Tool-Using Systems

Autonomous agents expand the red team surface beyond single-prompt testing: the objective can be reached across a multi-step loop (planner → tool selector → tool executor → memory), through a poisoned tool or MCP server, or via accumulated context across turns. Testing must therefore include multi-turn and cross-tool attack paths, not just single-input attempts — and the environment must be isolated so that a successful agentic attack during testing cannot cause real-world action (e.g., a live financial transaction). Sandbox isolation is itself part of the Rules of Engagement, not an afterthought.

3.7 Safety and Legal Handling (Non-Negotiable)

- **Authorization first.** Testing an AI system without explicit, documented authorization may be unlawful and is always out of policy. The RoE is the artifact that makes an engagement legitimate.
- **Isolated environments.** Agentic and tool-enabled testing runs in sandboxes with no path to real external action or production data, unless production testing is explicitly authorized with compensating safeguards.
- **Sensitive-capability findings.** Where testing probes high-harm categories (e.g., CBRN uplift), results are handled under strict disclosure controls: minimal reproduction detail in general reports, restricted distribution, and no publication of uplift-enabling specifics.
- **Data handling.** Prompts and outputs captured during testing may contain sensitive or personal data; they are stored, redacted, and retained under the same evidence-grade discipline as any other security evidence.

3.8 Team Composition

The NIST guidance that red team quality depends on the team's background, expertise, and diversity (Section 10) translates into concrete roles. A capable engagement typically draws on:

- **Red team lead** — owns the RoE, methodology, and final report; accountable for reproducibility discipline
- **AI/ML specialist** — understands model behavior, alignment, and evaluation, and can reason about *why* an attack works
- **Application/infrastructure security tester** — covers the implementation and infrastructure assessment areas
- **Domain/subject-matter expert(s)** — provide the context to test harm categories the security team cannot judge alone (e.g., legal, medical, financial, safety); this is where demographic and interdisciplinary diversity directly improves coverage
- **Blue team / detection liaison** — connects findings to monitoring and response (see purple teaming below)

A team homogeneous in background will systematically miss harm categories outside its own experience — a coverage gap, not a skill gap.

3.9 Purple Teaming

Red teaming (offense) and blue teaming (detection and response) deliver more value combined than in isolation. In a purple-team model, red team findings are shared with the detection team in near-real time so that each attack path is checked not only for *whether it succeeds* but for *whether it is detected and responded to* — directly linking red team output to the monitoring capability (see ODA3-2026-07-CHT-SEC-002). A finding that succeeds *and* goes undetected is more severe than one that succeeds but triggers an alert; purple teaming is how that distinction gets measured rather than assumed [T2].

3.10 Scoping and Sizing

The most common planning failure is an open-ended engagement with no coverage target. A defensible scope states, before testing: which of the four assessment areas are in play; which harm categories from the risk catalog are in and out; the attack surfaces and lifecycle stages covered; the manual/automated split; and the stopping condition (time-boxed, coverage-based, or objective-based). Effort scales with capability — a read-only Q&A system may need days of largely automated coverage, while an autonomous agent with external actions warrants weeks of expert-led testing plus automated regression. Under-scoping produces a report that looks complete but tested a fraction of the surface; the antidote is an explicit, signed-off coverage statement, not a longer engagement alone.

---

4. Practitioner Decision Framework

4a. Depth by System Risk

Risk Factor	Low	Medium	High	Critical
System capability	Read-only Q&A	RAG over internal content	Tool-using assistant	Autonomous agent with external actions
Data/harm exposure	Public, low-harm	Internal, confidential	Regulated data	CBRN/safety-critical or high-value action
Recommended cadence	Pre-release + annual	Pre-release + semi-annual	Pre-release + quarterly + on change	Continuous + pre-release gate + change-triggered
Testing mix	Automated coverage	Automated + targeted manual	Automated + expert manual	Expert-led + automated regression + specialist harm review

4b. Decision Tree

```
Is the system authorized for testing with a signed RoE?
  NO  → Stop. Obtain authorization first. No testing proceeds.
  YES → Does it take autonomous or external actions?
        NO  → Does it retrieve untrusted/external content?
              NO  → Model + implementation testing; automated
                    coverage + targeted manual.
              YES → Add indirect-injection and RAG-poisoning
                    objectives; test the retrieval path.
        YES → Mandatory isolated sandbox. Add multi-step/agentic
              and tool/MCP objectives. Expert-led manual required;
              automation for regression.
```

4c. Prioritization

Scenario	Risk	Priority	Timeframe
High-capability agent shipped with no red team	Critical	Highest	Before further rollout
Public LLM app never tested for injection/jailbreak	High	Immediate	Pre-release / 30 days
Findings reported but never retested after fixes	High	Immediate	30 days
Automated coverage only; no expert testing on high-stakes system	Medium	High	60 days
No mapping of findings to a risk catalog (NIST/OWASP)	Medium	Medium	Next cycle

5. Red Team Controls & Practices Matrix

ID	Practice	Effect.	Complexity	NIST AI RMF	OWASP Anchor	GAISSF™ Domain	UAIF™ Category
R-01	Documented Rules of Engagement before any testing	High	Low	Govern	—	Assurance Governance	Event Logging
R-02	Threat model mapped to NIST AI 600-1 risk categories	High	Med	Map	LLM/Agentic Top 10	Risk Evaluation	Threat Assessment
R-03	Isolated sandbox for agentic/tool testing	High	Med	Manage	ASI02	System Architecture	Privilege Abuse
R-04	Full attempt logging (input, output, model version, outcome)	High	Low	Measure	—	Evidence Collection	Event Logging
R-05	Reproducibility confirmation before a finding is filed	High	Med	Measure	—	Validation	Incident Detection
R-06	Impact × exploitability severity scoring	High	Low	Measure	—	Risk Evaluation	High Severity
R-07	Manual/expert testing on high-stakes systems	High	High	Measure	—	Validation	Threat Simulation
R-08	Automated coverage + regression campaigns	Med–High	Med	Measure	—	Continuous Monitoring	Threat Simulation
R-09	Judge-model calibration documentation	Med	Med	Measure	—	Response Validation	Output Manipulation
R-10	Findings mapped to controls with named owners	High	Low	Manage	—	Access Governance	Incident Detection
R-11	Remediation retest (closure, not suppression)	High	Med	Manage	—	Validation	Incident Detection
R-12	Disclosure handling for sensitive-capability findings	High	Med	Govern	—	Confidentiality	Information Disclosure
R-13	Purple teaming: findings checked for detection/response, not just success	Med–High	Med	Manage	—	Continuous Monitoring	Incident Detection

NIST AI RMF references indicate the primary function (Govern/Map/Measure/Manage). GAISSF™ and UAIF™ mappings reference GAISSF™ v1.0 and UAIF™ v1.0 respectively, as factual conformance-alignment statements under GEL v1.0 Section 10.3. OWASP Agentic Top 10 IDs (ASI-series) are cited only where an anchor is verified; per-technique MITRE ATLAS IDs are intentionally omitted pending verification against the current published matrix. EU AI Act adversarial-testing obligations for general-purpose models with systemic risk are noted at the framework level but not mapped to specific articles pending verification against the consolidated text.

6. Reporting: What a Defensible Finding Contains

A finding that an assessor or a second red team can trust includes, at minimum:

- **Objective** — what the attack was trying to achieve (the harm), not just the technique
- **Reproduction reference** — enough for an authorized tester to re-trigger it, held at the appropriate sensitivity level (full detail in restricted annex where the capability is dangerous)
- **Target identity** — exact model, version, prompt/config, and environment, since a finding on one version may not hold on another
- **Reliability** — how often the attack succeeded (with sample size), not a single lucky success
- **Severity** — impact (blast radius) × exploitability (ease, prerequisites, reliability)
- **Mapped control(s)** — the recommended mitigation and the responsible owner
- **Status** — open / mitigated / retested-closed, updated through the remediation cycle

A report that lists clever attacks without severity, reproducibility, version pinning, and control mapping is a demonstration, not an assurance artifact.

7. AI Lifecycle Mapping

Phase	Red Team Concern	Activity
Model Acquisition/Selection	Inherited weaknesses of a third-party model	Baseline model-evaluation red teaming before adoption
Fine-tuning	Alignment regression introduced by tuning	Re-test refusal/jailbreak resistance post-tuning
Evaluation	Coverage of harm categories before release	Full engagement mapped to risk catalog; release gate
Deployment	Implementation and infrastructure exposure	Implementation + infrastructure testing on the deployed config
Inference/Runtime	Live multi-step and agentic abuse	Runtime behavior analysis; agentic multi-turn testing
Monitoring	Drift and newly discovered attack classes	Periodic regression campaigns; re-test on model/prompt change
Retirement	Residual exposure of decommissioned systems	Confirm access revocation; retain findings evidence per policy

8. Assessment & Assurance Lens

8a. Verification vs. Validation of a Red Team Program

Aspect	Verification	Validation
Question	Was the engagement conducted as documented?	Did it actually find the failures that matter?
Evidence	RoE, test plan, attempt logs, coverage report	Reproduced findings, severity distribution, retest closure rate, escaped-defect history
Failure mode if skipped	Untraceable, non-repeatable testing	A program that runs on schedule but never surfaces real risk

8b. Assurance Questions by Role

Security Architect: How is coverage across the four assessment areas decided and documented? What is in scope and explicitly out of scope, and why?

Red Team Lead: How is reproducibility confirmed before a finding is filed? How is the judge model calibrated, and what are its error rates?

AI Governance Lead: How are findings mapped to a risk catalog (NIST AI 600-1 / OWASP), and how does the program feed the risk register? How are sensitive-capability findings handled?

CISO: What is the retest closure rate, and are fixes confirmed or merely assumed? What high-harm categories has the program *not* tested, and is that acceptance documented?

8c. Residual Risk

Red teaming reduces but never eliminates unknown-unknown risk. Residual risk includes: attack classes not yet invented at test time; harm categories consciously scoped out; the gap between sampled worst-case and true worst-case behavior; judge-model blind spots; and the reality that a passed engagement is evidence of *tested* robustness, not proof of safety. A red team report should state what it did not test as clearly as what it did — absence of findings in an untested area is not assurance.

---

9. Maturity Model

Level	Characteristics
1 — Ad Hoc	Occasional informal "try to break it" sessions; no RoE, no reproducibility, no mapping
2 — Repeatable	Documented RoE and test plans; findings recorded consistently
3 — Managed	Coverage mapped to a risk catalog; severity scoring; findings mapped to controls and owners
4 — Measured	Retest closure tracked; automated regression + expert testing; judge calibration documented
5 — Optimized	Continuous and change-triggered testing integrated into the SDLC; coverage and escaped-defect metrics trended
6 — Assured	Independent assurance demonstrates program efficacy with objective evidence; sensitive-capability handling meets external standards

Getting Started (Crawl → Walk → Run)

For organizations beginning from zero, the maturity model above is a destination, not a starting instruction. A pragmatic sequence:

- **Crawl:** Pick one high-risk system. Write a Rules of Engagement. Run automated coverage against the OWASP LLM Top 10 with a basic finding template. Goal: a first defensible, reproducible finding set.
- **Walk:** Add expert manual testing on that system; map findings to NIST AI 600-1 categories; introduce severity scoring and remediation retest. Extend to the next few systems by risk rank.
- **Run:** Integrate testing into the SDLC as a release gate; add continuous and change-triggered campaigns; introduce purple teaming and coverage/escaped-defect metrics. Goal: a measured, always-on program rather than periodic events.

---

10. Research Insight ★

Finding: Verified standards convergence — OWASP's GenAI Red Teaming Guide organizes testing across four areas (model evaluation, implementation testing, infrastructure assessment, runtime behavior analysis), while NIST AI 600-1 defines twelve generative-AI risk categories and four red-teaming approaches (General Public, Expert, Combination, Human/AI), explicitly noting that red team quality depends on the team's background, expertise, and demographic and interdisciplinary diversity [T1].

Technical Significance: These two independently developed frameworks agree on the core principle: effective AI red teaming is a *structured, coverage-driven process*, not a talent-dependent hacking event. The emphasis on team diversity is itself a technical finding — homogeneous teams systematically miss harm categories outside their own experience.

Validation of Guidance: This supports the engagement-lifecycle discipline (Section 3.3), the four-area coverage requirement (Section 3.2), the risk-catalog mapping practice (R-02), and the manual/expert requirement for high-stakes systems (R-07).

Operational Lesson: Build the program around documented coverage and reproducibility, staff it for diversity of expertise, and treat any harm category the team cannot competently test as an explicit, owned coverage gap — not a silent omission.

11. Common Mistakes

Mistake	Correct Approach
Testing without documented authorization	No engagement begins without a signed Rules of Engagement
Treating red teaming as a one-time event	Run it as a repeatable, change-triggered process across the lifecycle
Confusing red teaming with model evaluation	Evaluation measures average case; red teaming targets adversarial worst case
Filing non-reproducible "wins" as findings	Confirm reproducibility with sample size before a finding is filed
No version pinning on findings	Record exact model/prompt/config/environment; findings are version-specific
Automation only, on high-stakes systems	Add expert manual testing; automation misses novel and multi-step attacks
Ignoring judge-model calibration	Document judge error rates; an uncalibrated judge corrupts every ASR number
Reporting attacks without severity or control mapping	Score impact × exploitability; map each finding to a control and owner
Assuming a fix works without retest	Retest the attack path to confirm closure, not suppression of one variant
Publishing or over-detailing dangerous-capability findings	Handle sensitive findings under disclosure controls; restrict uplift-enabling detail
Reporting only what was found, not what was skipped	State scope and untested areas explicitly; unfound ≠ safe

12. Notably Absent

Current publicly available evidence does not include several things a red team program is often assumed to have [T1/T2/T4]:

- **No universally accepted, quantitative AI red teaming benchmark** — there is no standard, cross-system measure of "how thoroughly was this tested" or "how robust is this model," so coverage and ASR figures are not comparable across organizations or vendors
- **No ratified certification standard specifically for AI red team competency or program maturity** — maturity models (including the one above) are structural guidance, not accredited criteria
- **No settled methodology for measuring the worst-case/true-worst-case gap** — a passed engagement bounds tested behavior, not all possible behavior, and the size of that gap is not currently quantifiable
- **No standardized safe-handling protocol for dangerous-capability (e.g., CBRN-uplift) findings** accepted across the field — disclosure practice remains organization-specific

Organizations should therefore treat red team results as bounded, comparative-only-within-their-own-program evidence, and should be skeptical of vendor claims of "red-team tested" absent a stated scope, method, and coverage.

---

13. Quick Reference

Engagement Readiness Check

If you cannot answer this…	You are not ready to test
Is there a signed Rules of Engagement?	Authorization gate
Which of the 12 NIST harm categories are in/out of scope?	Coverage definition
Is agentic testing sandboxed from real action?	Safety isolation
How will a "success" be scored, and by what judge?	Success criteria
How will findings be stored and sensitive ones restricted?	Disclosure handling

Program Checklist

Item	Yes	No	Partial
Signed RoE precedes all testing	☐	☐	☐
Threat model mapped to a risk catalog	☐	☐	☐
Agentic testing runs in an isolated sandbox	☐	☐	☐
Every attempt logged for reproducibility	☐	☐	☐
Findings carry severity + version + reliability	☐	☐	☐
Findings mapped to controls with owners	☐	☐	☐
Remediation retest confirms closure	☐	☐	☐
Sensitive-capability disclosure handling defined	☐	☐	☐
Untested areas documented in every report	☐	☐	☐

---

14. If You Can Only Do Five Things This Week

#	Action	Impact	Effort
1	Require a signed Rules of Engagement before any AI testing — no exceptions	Very High	Low

#	Action	Impact	Effort
2	Map your next engagement’s scope to the NIST AI 600-1 risk categories, and record what is out of scope	High	Low
3	Ensure any agentic/tool testing runs in an isolated sandbox with no path to real action	Very High	Medium
4	Adopt a finding template that forces severity, version pinning, reproducibility, and control mapping	High	Low
5	Retest one previously "fixed" finding to confirm it is actually closed	High	Low

For Leadership: Questions to Ask Your Team

Do we red team our AI systems on a schedule, or only after something goes wrong?

Can we produce reproducible, control-mapped findings — or just anecdotes?

Which high-harm categories have we deliberately *not* tested, and who accepted that risk?

When we fix a red team finding, do we confirm the fix, or assume it?

15. References

Source	Relevance	Tier
OWASP GenAI Red Teaming Guide (four areas: model evaluation, implementation testing, infrastructure assessment, runtime behavior analysis)	Basis for Sections 3.2, 3.3, Research Insight	T1
NIST AI 600-1, Generative AI Profile (twelve risk categories; four red-teaming approaches: General Public, Expert, Combination, Human/AI; team-diversity guidance)	Basis for Sections 3.4, 3.5, 10	T1
OWASP Top 10 for LLM Applications / OWASP Top 10 for Agentic Applications (2026)	Risk/technique anchors for threat modeling and Controls Matrix	T1/T2
MITRE ATLAS (adversarial technique taxonomy for AI systems)	Structuring attack approaches; technique IDs to be verified per-use	T2
EU AI Act (adversarial-testing obligations for general-purpose models with systemic risk)	Framework-level regulatory driver	T2

Framework counts and technique IDs change between revisions; verify against the primary published source before citing in external certification materials. Specific MITRE ATLAS technique IDs and EU AI Act article numbers are intentionally omitted pending direct verification against current published matrices and the consolidated regulation text, consistent with this document’s evidence-fabrication safeguard. Additional externally suggested inclusions reviewed for this edition and excluded: specific ATLAS technique-ID-to-activity mappings beyond AML.T0051 (unverified this cycle), a specific EU AI Act GPAI systemic-risk article citation (plausible but unverified), and specific attack-success-rate percentages (no sourced origin). Named open-source tooling (PyRIT, Garak, Promptfoo) is included as illustrative example rather than endorsement.

Bottom Line

AI red teaming earns its assurance value through process, not showmanship. Authorize it, scope it against a real risk catalog, sandbox anything that can act, log everything so findings are reproducible, score by impact and exploitability, map to controls and owners, and retest to confirm closure. A program built this way produces evidence an assessor can trust; a program built around clever one-off attacks produces stories. Test worst case, document what you did not test, and never let "no findings" be mistaken for "safe."

Document Metadata

Field	Value
Classification	Public
Intended Audience	Primary: CISOs, Security Architects, AI Governance Leads, Compliance Officers. Secondary: CEOs, CFOs, Board Members, General Counsel, Risk Committees
Evidence Confidence	Mixed T1–T4; see inline tags
Review Cycle	Quarterly
ODA3 Framework Cross-References	GAISSF™ v1.0 (Assurance Governance, Validation, Risk Evaluation, System Architecture domains), UAIF™ v1.0 (Threat Simulation, Threat Assessment, Incident Detection categories)
Related ODA3 Publications	ODA3-2026-07-CHT-SEC-001 — Prompt Injection Mitigation Controls; ODA3-2026-07-CHT-SEC-002 — AI Monitoring Architecture
Framework Licensing	GAISSF™, UAIF™, and AI-IRF™ are frameworks of ODA3 Institute, referenced under the GAISSF Ecosystem License (GEL) v1.0. Mappings herein are factual conformance-alignment statements (GEL v1.0 §10.3) and do not constitute certification.