PLAYBOOK Updated March 2026

# AI Red Teaming Playbook

Scope, phases, evidence, and metrics for testing LLMs, agents, and RAG—so security teams can run programmes that match how AI systems actually ship.

**Audience** Red teams, AppSec, ML platform ~8 min read

This playbook translates offensive security discipline to generative AI: scoping, methodology, tooling categories, and evidence collection for leadership and auditors. It complements the OWASP LLM Top 10 and draws on continuous-testing patterns described in Agentic Assure's internal security architecture research (multi-module orchestration, privacy probes, and regression gates).

## KEY TAKEAWAYS

- Authorise every engagement; agentic workflows need explicit blast-radius rules.
- Automate breadth, then use experts for chains and business-logic abuse.
- Ship regressions when the industry learns a new exploit class.

## 1. Objectives and scope

**Primary goals.** Discover instruction hijacking, unsafe tool use, data leakage, jailbreaks, and robustness failures before attackers or regulators do.

**Scope clearly.** Define model version, system prompts, tools/APIs, RAG corpora, data residency, and prohibited actions (e.g., no production data exfiltration, no disruption of live users).

**Rule of engagement.** Written authorisation, emergency contacts, and logging requirements are mandatory for any testing beyond synthetic environments.

## 2. Threat modelling snapshot

Start from assets (models, indices, credentials), entry points (chat UI, API, batch jobs, plugins), and adversaries (external users, compromised partners, malicious insiders). Map controls: authentication, RAG ACLs, output filters, human approvals.

- **STRIDE-style prompts for AI:** Spoofing user intent; Tampering with retrieved docs; Repudiation of agent actions; Information disclosure via outputs; Denial of wallet/service; Elevation via tool abuse.
- **Agentic surfaces:** Planner loops, memory stores, and cross-tool workflows multiply blast radius—test chained exploits, not only single-turn prompts.

## 3. Test phases

### Phase A — Baseline characterisation

Document intended policies, blocked topics, tool matrix, and golden-path behaviours. Capture non-security baselines (latency, token use) to compare under attack load.

### Phase B — Automated adversarial probing

Run libraries and harnesses that encode known attack classes: prompt injection fuzzing, jailbreak templates, encoding and multilingual obfuscation, policy-violation probes, and structured-output stress tests.

### Phase C — Manual creativity and chain building

Skilled testers explore logical gaps: indirect injection via uploaded PDFs, multi-session memory poisoning, and social engineering of the model's "helpful" bias.

### Phase D — Privacy and extraction

PII scanners on outputs; canary and completion-style prompts; membership-style probes where ethically justified. Align with LLM02 and training-data exposure risks.

### Phase E — Resilience and operations

Rate-limit bypass, token inflation, long-context abuse, and cost exhaustion scenarios (LLM10). Verify monitoring and circuit breakers.

## 4. Evidence and reporting

Each finding should include: severity, OWASP LLM mapping, reproduction steps (prompts redacted or hashed if sensitive), model/tool versions, timestamps, and suggested mitigations. Store artifacts in tamper-

evident systems where compliance requires it.

## 5. Tooling landscape (categories)

| CATEGORY | EXAMPLE USE |
| --- | --- |
| LLM vulnerability scanners / harnesses | Batched jailbreak and injection campaigns with pass/fail rubrics. |
| Adversarial ML (CV / tabular) | Evasion and robustness testing for non-text models. |
| Privacy tooling | PII detection, redaction validation, memorisation heuristics. |
| Red-team orchestration | CI/CD integrated suites, scheduling, dashboards. |

Tool names change quickly; evaluate against your stack and data-handling requirements rather than chasing a fixed list.

## 6. Integration with SDLC

- **Pre-release:** Gate promotions from staging with automated suites and sampled manual review.
- **Post-release:** Continuous monitoring on sampled live traffic; periodic full red-team sprints.
- **After incidents:** Add regressions for every novel exploit class discovered internally or in the wild.

## 7. Metrics that matter

- Attack success rate by category (injection, jailbreak, tool abuse) with trend lines per release.
- Mean time to remediate critical findings.
- Coverage: % of tools/prompt surfaces exercised.
- False positive burden on engineering—keep reports actionable.

---

**Agentic Assure** · [www.agenticassure.ai](http://www.agenticassure.ai)

Internal alignment: modular security, privacy, and quality testing pipelines as described in Agentic Assure product security documentation.