SISA PRISM · LAB & TESTING SERVICES

AI Agent Pen-Test

CISO Scoping & Report-Acceptance Checklist

Companion to “Pen-Testing AI Agents.” Two lists: what to ask for before the test, and what to verify in the report before you accept it. | June 2026

Use **Part A** when you write the RFP or scope of work — it makes sure the test actually covers the agent’s whole operating cycle. Use **Part B** when the report lands, before you sign it off. Each item ties back to the six-class AI-agent trap taxonomy: **content injection, semantic manipulation, cognitive state, behavioural control, systemic,** and **human-in-the-loop.** Tick what is covered; treat blanks as questions for your tester.

Part A — Before the test: what to ask for

A0. Foundations (get these right first)

- ☐ A full AI / agent asset inventory is completed before scoping — including shadow agents — and the scope lists every agent, its tools, its data stores, and its sub-agents.
- ☐ Crown-jewel objectives are defined: the specific data and actions an attacker must not be able to make the agent reach or perform.
- ☐ The environment is documented: every input channel, the tool / MCP catalogue, memory and RAG stores, sub-agent orchestration, and every human-approval point.
- ☐ Rules of engagement cover autonomous and irreversible actions: test data, sandboxes, a kill-switch, and explicit blast-radius limits.
- ☐ The methodology is human-led and AI-augmented, and objective-driven — the vendor does not claim “fully autonomous” complete coverage.
- ☐ Findings will be mapped to recognised frameworks: the six-class trap taxonomy, OWASP Agentic & LLM Top 10, MITRE ATLAS, and NIST AI RMF.
- ☐ A retest of remediated findings is included in scope and price.

A1. Content injection — perception layer

- ☐ The test covers every parse layer, not just the rendered view: HTML comments, off-screen / zero-opacity CSS, ARIA and accessibility attributes, image steganography and metadata, Unicode tricks (right-to-left override, zero-width characters), and alternate encodings (Morse, base64).
- ☐ The test exercises every untrusted input channel the agent reads: web pages, documents, emails, calendar invites, support tickets and issues, user reviews / user-generated content, and API responses.

A2. Semantic manipulation — reasoning layer

- ☐ The test probes decision integrity: can framing, authoritative-language saturation, or position-in-context steer the agent's conclusions, scores, or recommendations?
- ☐ The test includes long-context robustness (the "lost-in-the-middle" effect) for agents that process large documents.

A3. Cognitive state — memory layer

- ☐ The test attempts RAG / knowledge-base poisoning at realistic low contamination levels.
- ☐ The test includes persistence: does a single act of poisoning survive a session reset and re-activate in a later, unrelated interaction (latent memory)?
- ☐ The test covers adversarial few-shot / example poisoning in the agent's context.

A4. Behavioural control — action layer

- ☐ The test is objective-driven: it attempts to drive the agent to exfiltrate specific data, misuse a tool, write or delete files, execute commands (RCE via tool chains), and spawn sub-agents.
- ☐ The test verifies least-privilege of every tool and credential the agent can reach — not just whether a tool can be called, but what it can be made to do.

A5. Systemic — multi-agent & supply chain

- ☐ The test probes inter-agent trust and orchestration: can a low-privilege agent recruit or escalate through a higher-privilege one?
- ☐ The test covers the shared supply chain: MCP servers and SDKs, skill / instruction files, and package dependencies.
- ☐ The test attempts compositional (fragmented) payloads that are benign in isolation but reassemble when the agent aggregates them.

A6. Human-in-the-loop — the supervisor

- ☐ The test attacks the approval UX: can the agent's own summaries be made misleading enough to win approval for an action a human would reject if described accurately?
- ☐ The test specifically targets deceptive summarisation and approval-fatigue exploitation.

Part B — After the test: what to verify in the report

A credible AI-agent pen-test report is not a list of model misbehaviours. It is a record of **demonstrated actions against defined objectives,** with a clear statement of coverage. Before you accept the report, confirm each of the following.

- ☐ Findings are mapped to the six-class taxonomy and a recognised framework — and the report includes a coverage statement listing what was and was not tested.
- ☐ Each finding states the objective achieved — the actual unauthorised action taken or data reached — not just "the model responded to a test case."
- ☐ Every finding has a reproducible proof-of-concept: the exact trap payload, the input channel used, and the agent's action trace (which tools were called, which data was touched).
- ☐ There is at least one memory-persistence result, or an explicit statement that persistence was tested and none was found.
- ☐ Tool / MCP attack-surface coverage is explicit: which tools and credentials were reachable, and which were actually abusable.
- ☐ Multi-agent and orchestration findings are present — or the report explicitly notes the system is single-agent.
- ☐ Human-approval-layer findings are present — or the report explicitly notes the approval flows were tested.
- ☐ Severity is rated with agent-specific context — autonomy, blast radius, and reversibility — not a raw CVSS score alone.
- ☐ Remediation is agent-appropriate: input provenance and trust boundaries, least-privilege tool scoping, memory validation, guardrail tuning, and approval-flow redesign — not just "patch the server."
- ☐ Business impact is tied to your defined crown jewels.
- ☐ Retest evidence is provided for any findings already remediated.

Red flags — if you see these, push back

- ☐ "We tested prompt injection" with no tool, memory, or multi-agent coverage.
- ☐ Findings are model-output anecdotes with no demonstrated action and no trace.
- ☐ "Fully autonomous, 95%+ coverage" with no human-led validation.
- ☐ No coverage statement; no list of what was not tested.
- ☐ No persistence / memory testing at all.
- ☐ Generic CVSS scores with no autonomy or blast-radius reasoning.

SISA Prism. PrismDiscover builds the AI and agent asset inventory — including shadow agents — so the test is scoped against reality. PrismStrike runs the offensive validation through two modules: **GenAI Application Pen-Test** (LLM-integrated apps) and **Agentic Pen-Test** (agentic workflows, tool use, MCP attack surface, multi-step chains and sub-agents), objective-driven and mapped to the taxonomy above.

To walk this checklist through against your environment, talk to the SISA Prism team — sisa.ai.