Harness and Runtime Hardening — One-Page Checklist

Companion to Agentic AI Security Training — Chapter 6
Use this checklist when configuring the host, the harness, and the network around an agent.

Principle in one sentence

The harness is where Zero Trust is implemented in practice — it dispatches tool calls, assembles context, runs hooks, and writes audit; the host that runs it determines the agent's network reach, filesystem access, and blast radius — both deserve security treatment in their own right, not as undifferentiated infrastructure.

Network egress (the most effective single control)

- [ ] Default-deny outbound. The harness has no internet access by default.
- [ ] Outbound destinations on an explicit allow-list, configured per agent and per tool (e.g. `api.openai.com`, `slack.com`, news provider domain).
- [ ] Domain-based filtering at an application-layer proxy (Squid, mitmproxy in transparent mode, cloud-provider equivalent).
- [ ] DNS filtered at the resolver. (A compromised agent restricted at the proxy can still encode data into DNS lookups.)
- [ ] Internal services accessed only over a VPN (Tailscale, WireGuard, mTLS). No services exposed on the public internet.
- [ ] Agent holds no SSH credentials between hosts. Lateral movement is structural, not policy-based.

Filesystem isolation

- [ ] `read_file` / `write_file` accept paths only within an explicit working directory.
- [ ] Path-traversal (`..`) rejected at the tool boundary, not by the model.
- [ ] System directories mounted read-only; source code mounted read-only where possible.
- [ ] User config files excluded from the agent's filesystem view: `~/.ssh`, `~/.aws`, `~/.config/claude`, `~/.npmrc`, browser profile directories.
- [ ] Per-task ephemeral filesystem (microvm or container) so injection in one session does not carry to the next.
- [ ] Capabilities added deliberately, not removed reactively after an incident.

Hooks (inline guardrails)

- [ ] **PreToolUse: Bash** — match dangerous commands (`rm -rf`, `git push --force`, `curl ... | sh`); deny or escalate to HITL.
- [ ] **PreToolUse: Write** — block writes outside the working directory; block writes to dotfiles.

- [ ] **PreToolUse: WebFetch** — enforce the domain allow-list at hook time as a second layer.
- [ ] **PostToolUse: *** — redact known secret patterns out of tool outputs before they reach the model's context (see Secret Management checklist).
- [ ] **UserPromptSubmit** — pattern-strip known injection markers (Chapter 4 L3).
- [ ] Hooks are defence-in-depth; the PEP at the tool layer remains the hard authorisation point.

Process model

- [ ] One harness process per agent. No shared process between agents on the same host.
- [ ] Ephemeral subprocess per task for code execution / shell.
- [ ] No shared filesystem between agents.
- [ ] No shared credentials between agents (per-agent service accounts).

Resource limits

- [ ] Loop iteration cap per session.
- [ ] Wall-clock limit per session.
- [ ] Per-tool rate limits per session.
- [ ] Token and monetary cost budgets per session.
- [ ] OS-level memory and disk quotas.
- [ ] Outbound network quotas (bytes-per-session ceiling for exfiltration detection).

Audit and observability

- [ ] Every tool call logged: identity, tool, arguments (redacted), outcome, latency, cost.
- [ ] Audit log written to a store outside the agent's reach. (An agent with write access to its log can rewrite it.)
- [ ] Conversation transcripts logged with provenance tags.
- [ ] Logs streamed in real time, not batch-uploaded.
- [ ] Alerts on outliers: new tool used, tool calls > 99th percentile, off-hours invocation, new egress destination.
- [ ] Retention ≥ 90 days for general agents, longer for high-risk.
- [ ] Periodically replay an old session into a sandbox to verify the audit is reconstructable.

Anti-patterns to remove

- [ ] Running the agent on a laptop under the developer's user account.
- [ ] Exposing the harness's web interface on a public port.
- [ ] Disabling hooks for performance reasons.
- [ ] Storing audit logs inside the agent's working directory.
- [ ] Relying on the model to refuse dangerous commands.

- [ ] Allowing the agent to install its own packages at runtime.