Least-Privilege Agent Setup — One-Page Checklist

Companion to Agentic AI Security Training — Chapter 5
Use this checklist when adding capabilities to an agent or onboarding a new agent.

Principle in one sentence

An agent's authority is the authority delegated to it; least privilege applies along two dimensions — the scope of each tool, and the separation of roles across distinct agents — and is the principal lever for limiting the consequences of any successful compromise.

Per-tool scoping

- [ ] Begin from deny-by-default. Every agent starts with zero tools; each capability is added explicitly when the task requires it.
- [ ] For each tool, narrow the configuration as well as its presence:
- [ ] `slack_post_message` → channel allow-list, not arbitrary channel argument.
- [ ] `read_file` / `write_file` → working directory only; reject path-traversal (`..`) at the tool boundary.
- [ ] `web_fetch` → domain allow-list, not arbitrary URL.
- [ ] `git_commit` → single repository, write access limited to a specific branch.
- [ ] Prefer read-only over read-write where the task allows.
- [ ] Prefer idempotent operations over non-idempotent.
- [ ] Prefer reversible operations over irreversible. Irreversible actions (delete, payment, public post, transaction) get HITL.
- [ ] Shell access is the most difficult tool to constrain. If granted, the agent runs in a sandbox and every command passes through a PEP.

Per-role separation

- [ ] Different tasks performed by different agents. A coding agent does not need email; a research agent does not need a shell; a design agent does not need access to secrets.
- [ ] No "all-in-one assistant" with shell, email, Slack, database, and irreversible-action tools combined. (This combines the worst case of every threat class.)
- [ ] Hand-off between agents, not capability union. When agent A delegates to agent B, B operates with B's capabilities, not the union of both.
- [ ] Per-role identity. Each agent has its own service account, its own tokens, its own audit trail. (See Secret Management checklist.)
- [ ] Per-role guard prompts. Each agent's strict-ignore guard is tailored to its role — and explicitly enumerates tools the agent does not have.

Sandboxing risky tasks

- [ ] Code execution, package installation, untrusted-file processing, and shell-using agents → sandbox.
- [ ] Sandbox isolation level appropriate to risk:
- [ ] Container — fast, inexpensive; escape paths exist for capable adversaries.
- [ ] VM / microvm (Firecracker, Kata) — kernel-level isolation; appropriate when executing untrusted code.
- [ ] Ephemeral per-task — created, used, destroyed for each task; bounds cumulative impact.
- [ ] Long-lived credentials and persistent identity stay outside the sandbox; the work is sandboxed, the controls are not.

Multi-agent discipline

- [ ] Orchestrator routing decisions derive from the user's prompt — never from text produced by a sub-agent or fetched content.
- [ ] Pipeline stages (A → B → C) treat each inter-agent boundary with the same controls used for external input — guard prompt, provenance tag, scope check.
- [ ] Avoid shared-memory and broadcast patterns. Direct hand-offs contain errors; shared memory propagates them.
- [ ] Where shared memory is necessary, segregate by writer and tag by provenance.
- [ ] Sign hand-offs from A to B; verify the signature at B's PEP. Confused-deputy patterns then become visible at the trust boundary.

Anti-patterns to remove

- [ ] Granting shell access by default.
- [ ] Sharing OAuth tokens across multiple agents.
- [ ] Running the agent under the user's identity.
- [ ] Treating prompt content as a substitute for enforced scope.
- [ ] Treating the orchestrator as implicitly trusted.
- [ ] Deferring sandboxing on cost grounds.

L0 → L3 ladder snapshot

Level	Tool scope	Role separation	Sandbox	Audit
L0	”Whatever the model needs”	One agent does everything	None	Conversation only
L1	Tool allow-list per agent	Separate prompts	None	Conversation + tool calls
L2	Per-tool argument constraints	Distinct service accounts	Containers for shell-using agents	Tool calls in central store
L3	PEP at every call, HITL for high-risk	Hand-off only, no shared scope	Microvm per task, ephemeral	Tamper-evident, replay-capable