Model and Supply-Chain Hygiene — One-Page Checklist

Companion to Agentic AI Security Training — Chapter 8
Use this checklist when adopting a model provider, an MCP server, a plugin, a skill, or an agent file.

Principle in one sentence

The trusted computing base of an agent extends well beyond the harness — model provider, MCP servers, plugins, skills, agent files, tool descriptions, and the build pipeline all influence behaviour, and almost none of them are authored in-house; supply-chain hygiene is therefore a first-order concern.

Model-provider trust

- [ ] Pin model versions explicitly. No `model-latest` aliases for high-consequence agents.
- [ ] Treat upgrades as deliberate changes that re-trigger evaluation (especially injection-resistance evals).
- [ ] Read the provider's data-handling agreement. Default contracts may permit prompt logging or training use; default is not appropriate for high-sensitivity workloads.
- [ ] Where data sensitivity is high, prefer a self-hosted or zero-retention deployment of an open-weights model, accepting capability reduction.
- [ ] Provider-specific keys: narrow scope, short lifetime, audited.

MCP servers

- [ ] Pin specific versions of every MCP server. Disable auto-update.
- [ ] Inspect the tool descriptions a server registers — not only its README. The descriptions are part of the system prompt.
- [ ] Run untrusted MCP servers in a sandbox alongside the harness, with the same egress restrictions applied to the agent.
- [ ] Treat any MCP server not authored or reviewed in-house as part of the trusted computing base.
- [ ] Subscribe to release notifications; review changes before upgrading.
- [ ] Periodically re-validate tool descriptions against the deployed system prompt.

Plugins, skills, and agent files

- [ ] Pin plugin and skill versions.
- [ ] Treat agent files (`CLAUDE.md`, `AGENT.md`, `.cursor/rules`, `.aider.conf.yml`) as security-relevant configuration. Review changes; restrict who can modify; consider hash-pinning if loaded from a shared location.
- [ ] Scan plugin and skill bundles for embedded tokens, suspicious URLs, shell invocations.
- [ ] Prefer skill marketplaces with provenance: signed bundles, publisher identity, version history.

- [ ] Do not load agent files from synced cloud folders, network mounts, or auto-synced repositories.
- [ ] The agent itself does not write its own agent files. (First injection becomes persistent backdoor.)

Tool-description injection

- [ ] Generate tool descriptions in-house where possible.
- [ ] Inspect descriptions before deployment. Strip imperative-mood sentences directed at the model.
- [ ] Length-limit descriptions; reject descriptions containing imperative instructions.
- [ ] Where third-party descriptions must be used, label them clearly as third-party metadata in the system prompt.

Build and deploy

- [ ] Reproducible install steps (pinned dependencies, no auto-update at session start).
- [ ] CI builds with checksums on artefacts.
- [ ] Signed releases for high-value agents.
- [ ] SBOM (software bill of materials) for the agent and its dependencies.
- [ ] Attestation chain from source to deployed binary.

Onboarding a new MCP server — pre-adoption review

- [ ] Read the server's source code.
- [ ] Read every tool description it registers.
- [ ] Note its declared permissions and required scopes.
- [ ] Check publisher identity and prior releases.
- [ ] Test in a sandboxed harness with no production credentials.
- [ ] Pin the version in deployment configuration.
- [ ] Issue short-lived, scope-limited credentials via the harness, not via long-lived credentials on disk.
- [ ] Restrict the server's outbound network to the relevant API only.

Anti-patterns to remove

- [ ] Auto-updating MCP servers, plugins, or skills.
- [ ] Treating tool descriptions as documentation.
- [ ] Trusting provider default data-handling.
- [ ] Loading agent files from shared locations.
- [ ] Treating "open-source" as equivalent to "reviewed".
- [ ] Choosing dependencies on download counts alone.

L0 → L3 ladder snapshot

Level	Models	MCP / plugins	Skills / agent files	Build / deploy
L0	`model-latest`; default contract	Install whatever the docs suggest	Auto-load anything in `~`	No CI / unsigned artefacts
L1	Pin model version	Pin server / plugin versions	Review at commit	Reproducible install steps
L2	No-train clause; zero-retention if available	Allow-list of vetted servers; descriptions inspected	Marketplace with publisher identity	CI builds; checksums
L3	Self-hosted or contractually verified; second-source plan	Sandboxed MCP servers, signed releases	Signed skill bundles, hash-pinned files	Signed releases, SBOM, attestation