AI agent security checklist

Before an AI agent can merge a pull request, post a message or change a ticket, answer three questions in writing: what can it touch, where do its secrets live, and how will you check afterwards what it did. Copy this file into a ticket or a doc, put a name next to every section, and tick each box only when its pass condition is true.

Agent: _____ Rollout owner: _____ Date: _____

0. Name the risks you are signing off on

- [ ] **Overreach.** You have listed every action the agent can take that nobody asked it to take, and decided which ones are acceptable.
- [ ] **Leaked secrets.** You know every credential the agent can read, including the ones that are only on the machine it runs on.
- [ ] **Prompt injection.** You have listed every place the agent reads text written by someone outside the team (public issues, inbound email, support tickets, web pages, documents).
- [ ] **Honest mistakes.** You have picked the one action you could not undo in under an hour, and it is blocked or needs a second person.

1. What can it touch (permissions)

Pass when one person can say, for every connected tool, what the agent can read, what it can change, and who stops it.

- [ ] **Effective access is written down.** One table, three columns: the chat app's scopes, the credentials on the machine or sandbox where the agent runs, and every connected tool's token (MCP servers, CLIs, API keys in the environment). The agent can do anything any one of them allows.
- [ ] **Who can ask is decided per channel.** For every channel, DM and issue tracker it listens to: who may start work, and who can read the answer.
- [ ] **Shared access is a choice.** If teammates can start work that runs on one person's credentials, that person agreed to it and knows which tasks it covers.
- [ ] **Every action is in one of three tiers.** Read (allowed), reversible write such as a draft PR, a comment or a branch (allowed and logged), irreversible or external such as a merge to main, a production deploy, a payment, a deleted record or a message to a customer (blocked or behind a human approval).
- [ ] **Tier three is enforced by the destination system.** Branch protection, required reviews, read-only database roles, scoped API keys and spend caps. A system prompt or custom instruction is guidance, never a control.
- [ ] **Untrusted text and sensitive tools are not in the same run.** An agent that reads public issues or inbound email does not also hold write access to private repositories or customer data in that run.
- [ ] **The tool's own permission settings are set.** Deny rules for secret files and dangerous commands, the sandbox and network defaults, and auto-approve modes turned off where policy requires it.

2. Where do the secrets live (credentials)

Pass when every credential the agent can reach has an owner, a scope, an expiry and a rotation step someone has rehearsed.

- [ ] **Inventory by machine.** For each machine or sandbox the agent runs on: whose account is it, which provider login, which Git credentials, which cloud CLI sessions, which `.env` files and which MCP configuration files exist there.
- [ ] **No personal admin tokens.** The agent uses its own scoped credential (a GitHub App or fine-grained token limited to named repositories, a read-only database user, a restricted API key), never a human's broad token.
- [ ] **Every token expires.** The expiry date is written next to the token in the inventory.
- [ ] **Secrets are out of prompts, memory and logs.** No keys pasted into chat, saved to agent memory, written into instructions files or echoed into tool output.
- [ ] **MCP and environment files are checked.** MCP configuration files and `.env` files on the agent's machine are not committed, and they hold only the credentials the agent needs.
- [ ] **Rotation is rehearsed.** You have rotated one agent credential end to end, and you know how long it took.
- [ ] **Revocation is one step.** Someone other than the machine owner knows how to disconnect the agent's machine and revoke its tokens.

3. How will you check afterwards (audit trail)

Pass when you can pick any change the agent made last week and trace it to the person who asked, in under ten minutes.

- [ ] **The join key exists.** For every agent action you can find: who asked, where (thread, issue or PR), which run, which machine, which credential, and what changed.
- [ ] **You know what each destination logs.** Each destination's audit log records the credential's owner, not the person who asked. You know which plan your audit log needs and how long it keeps events.
- [ ] **Agent-made changes are labeled.** Commits, PRs, comments and issues the agent creates carry a bot identity, a trailer or a label.
- [ ] **Retention covers your review window.** Agent run records, destination audit logs and chat history are kept at least as long as your incident review window.
- [ ] **One replay has been done.** You have traced one real agent action from the request to the change, start to finish.
- [ ] **The off switch works.** Someone who is not an admin has stopped a running task, and you know what it leaves behind.

4. Before you widen access

- [ ] All boxes above are ticked or have a dated exception with an owner.
- [ ] One thing changes per expansion: a new channel, a new tool, a new tier of action or a new group of people.
- [ ] The checklist is reviewed again after every change to tools, credentials or who can ask.