FREE KIT

AI Agent Guardrails Kit

Practical checklists for small businesses and solo builders

Keith Motte

30 years in Corporate IT. Last 7 at AWS. Built for SoCal small businesses.

TopofmindAI Builders · SoCal small business focus

Plain-English checklists · No hype

Topofmind.AI

@kman62

HOW TO USE

How to use this kit

- Print it or keep it next to your agent setup docs.
- Work the checklists in order the first time. After that, reuse the ones that match the job.
- Check a box only when the control is in place, not when you plan to add it.
- Use the risk-tier worksheet before you give an agent write, send, pay, or delete powers.
- Hand the contractor page to anyone who asks for logins before they touch your systems.
- Read the AI data hygiene callout when you train, fine-tune, or feed RAG corpora to an agent.
- This kit is practical security hygiene. It is not legal, compliance, or HIPAA advice. For medical or dental practices, loop in your privacy officer or counsel before you put PHI in any agent.

What this kit covers

Amazon Bedrock Guardrails and Agents, Claude / Claude Code, OpenAI Agents SDK and Codex, plus OWASP agent guidance and NSA/CISA/FBI CSI AI Data Security. Features named below come from those primary docs.

Audience: small-business owners and solo builders who use coding agents (Claude Code, Codex, Cursor) or business agents (Bedrock Agents, OpenAI Agents SDK).

AT A GLANCE

Guardrails at a glance

Where each control lives. Only features confirmed in primary docs.

Control	AWS Bedrock	Claude / Claude Code	OpenAI / Codex
Content / safety filters	Bedrock Guardrails content filters (Hate, Insults, Sexual, Violence, Misconduct, Prompt Attack)	Permission modes + deny/ask rules; classifier in auto mode	Safety classifier, moderation, input/output guardrails
Denied topics / scope	Denied topics policy	CLAUDE.md + deny rules + plan mode	Relevance classifier; agent instructions
Word / keyword blocklists	Word filters (custom words/phrases, profanity option)	Deny rules on tools/commands	Rules-based protections (blocklists, regex)
PII / sensitive data	Sensitive information filters (block or mask); many built-in PII types + regex	Read deny rules for .env/secrets; redact before logging	PII filter on outputs; tool output guardrails
Grounding / hallucination checks	Contextual grounding checks; Automated Reasoning checks	Give Claude a pass/fail check (tests, Stop hooks)	Output validation; output guardrails
Human approval for actions	User confirmation (CONFIRM/DENY); Return of control (RETURN_CONTROL)	Manual / ask rules; needs human for high-impact; plan mode	needs_approval / needsApproval; interruptions + resume
Least privilege identity	Agent service role + Lambda resource policy scoped to agent ARN; ApplyGuardrail	Allow/ask/deny; sandbox FS/network; no bypassPermissions on host	Scoped tools; sandbox_mode + approval_policy (Codex)
Sandbox / isolation	Separate accounts/roles; Lambda action groups; no prod creds in agent role	Bash sandbox (/sandbox); containers for bypassPermissions	Codex read-only / workspace-write; network off by default
Data provenance / integrity	Encrypt + classify agent data; hash RAG corpora; Bedrock PII block/mask	Deny secret paths; redact logs; isolate memory	PII filter; tool output guardrails; treat web results as untrusted
Tripwires / hard stop	Blocked-message response from Guardrails; deny confirmation	Deny rules; dontAsk; kill session / revoke keys	InputGuardrailTripwireTriggered / OutputGuardrailTripwireTriggered

CHECKLIST

Before you give an agent access

Identity and least privilege first. No agent gets prod keys on day one.

- [ ] Create a dedicated identity for the agent (IAM role, service account, or API key). Do not reuse your personal admin login.
- [ ] Grant least privilege only. List the exact APIs, buckets, repos, and tools the job needs. Drop everything else.
- [ ] Scope API keys by product and environment. Separate keys for sandbox, staging, and production.
- [ ] Use a separate cloud account or workspace for experiments when you can. Keep prod out of the blast radius.
- [ ] Never put production credentials in the agent prompt, CLAUDE.md, AGENTS.md, or chat history.
- [ ] Store secrets in a vault or secrets manager. Deny the agent Read on .env, .pem, .key, and secret paths.
- [ ] For Amazon Bedrock Agents, attach a custom service role with SourceAccount and SourceArn conditions. Replace wildcards with specific agent IDs after create.
- [ ] Scope Lambda resource-based policies so only that agent ARN can invoke action-group functions.
- [ ] If you attach a Bedrock Guardrail, grant bedrock:ApplyGuardrail on that guardrail ARN only.
- [ ] Turn on short-lived credentials where the platform supports them. Rotate keys on a calendar.
- [ ] Document who owns the agent identity and who can approve privilege changes.
- [ ] Test access with a dry run that should fail when the agent reaches for something outside scope.

12 items

CHECKLIST

Input and prompt-injection defenses

Treat every external string as data, not as instructions. That includes your own repo files.

- [ ] Treat user messages, emails, PDFs, web pages, ticket text, and tool outputs as untrusted data.
- [ ] Separate instructions from data with clear delimiters or a dedicated validation step before the main agent runs.
- [ ] On Bedrock, enable the Prompt Attack content filter and set strength for your use case.
- [ ] On OpenAI Agents SDK, add input guardrails. Prefer blocking mode (run_in_parallel=False) when a tripwire must stop tools from running.
- [ ] Do not pipe raw untrusted content into a coding agent. Summarize or sanitize first.
- [ ] Run the 60-second AGENTS.md check before you open a new repo (see coding-agent checklist).
- [ ] Review CLAUDE.md, .claude/settings.json, .mcp.json, and Copilot instruction files the same way you review code.
- [ ] Block or ask on network fetch tools (curl, wget, WebFetch) unless the domain is on an allowlist.
- [ ] For Codex, keep web_search on cached when you can. Treat live web results as untrusted.
- [ ] Disable or tightly scope MCP servers you did not install yourself.
- [ ] Reject instructions hidden in retrieved RAG chunks that try to change goals, exfiltrate data, or skip approvals.
- [ ] Log and alert when an input guardrail tripwire fires.

12 items

CHECKLIST

Output and action controls

Reads can be cheap. Writes, sends, payments, and deletes need a human.

- [ ] Classify every tool as low, medium, or high risk (OpenAI tool-safeguard pattern). High risk always pauses.
- [ ] Require human approval before send email, post to social, charge a card, wire money, delete data, change IAM, or deploy to prod.
- [ ] On Bedrock Agents, enable user confirmation (CONFIRM/DENY) on mutating action groups.
- [ ] Use return of control (RETURN_CONTROL) when your app must validate or edit parameters before execution.
- [ ] On OpenAI Agents SDK, set needs_approval=True (or needsApproval) on sensitive function tools and resume from saved state after review.
- [ ] Allowlist destinations for email, webhooks, and payment endpoints. Deny the rest.
- [ ] Set rate limits and spend caps per agent session (token, dollar, and tool-call budgets). Unbounded loops are a denial-of-wallet risk.
- [ ] Validate outputs against a schema before the next tool runs.
- [ ] Preview the exact action (tool name, target, parameters) on the approval screen. Do not rubber-stamp.
- [ ] Make high-impact actions idempotent when you can. Require a fresh approval if parameters change.
- [ ] Fail closed if the approval service, policy lookup, or audit log is down.
- [ ] Keep a kill switch that revokes the agent role and API keys in one step (see monitoring checklist).

12 items

CHECKLIST

Data and PII protection

Minimize what the agent can see. Mask what it must see. Keep PHI out of casual tools. CSI AI Data Security habits below.

- [ ] Inventory what data the agent can read: CRM, email, files, EHR, billing. Cut anything it does not need.
- [ ] Classify data by sensitivity and apply matching access controls. Treat agent output at the same classification as the input (NSA/CISA/FBI CSI AI Data Security).
- [ ] Track provenance for training, fine-tune, and RAG corpora: where each file came from and who changed it.
- [ ] Verify integrity with checksums or cryptographic hashes when you store or move AI datasets.
- [ ] Encrypt data at rest (AES-256 class) and in transit (TLS). Prefer vault-backed keys over keys in code.
- [ ] Use privacy-preserving techniques where practical: masking, redaction, or anonymization before the agent sees raw PII.
- [ ] On Bedrock Guardrails, enable sensitive information filters. Choose Block or Mask per field type.
- [ ] Remember Bedrock PII filters do not cover tool-call arguments, tool results, or tool definitions. Add your own checks there.
- [ ] Redact SSN, card numbers, passwords, and API keys from logs and transcripts.
- [ ] For medical and dental front desks: treat appointment notes, insurance IDs, and charts as PHI. Do not paste PHI into consumer chatbots. Use a BAA-covered path or keep PHI out of the agent. This is operational hygiene, not legal advice.
- [ ] Delete or sanitize retired AI data and media using a secure method (NIST SP 800-88 class practices), not a casual trash empty.
- [ ] Turn off prompt logging to vendors when your contract or policy requires it.
- [ ] Run ongoing data-security risk reviews (NIST RMF / AI RMF style) for agents that hold customer data.
- [ ] Run a quarterly access review of which agents still need customer data.

14 items

CHECKLIST

Coding-agent setup

Claude Code, Codex, Cursor, and friends. Sandbox first. Deny rules second. Bypass modes only in disposable boxes.

- [ ] Start in Manual (default) or plan mode for unfamiliar repos. Do not jump to bypassPermissions on your laptop.
- [ ] Enable the Claude Code Bash sandbox (/sandbox) with filesystem and network limits. Pair it with permission rules.
- [ ] Add deny rules for Read on .env, secrets/**, *.pem, and production kubeconfigs.
- [ ] Add ask or deny rules for git push, cloud deploy, and rm -rf style commands.
- [ ] For Codex, prefer sandbox_mode = workspace-write and approval_policy = on-request. Keep network off unless required.
- [ ] Never use Codex --dangerously-bypass-approvals-and-sandbox (--yolo) or Claude bypassPermissions outside a container or VM.
- [ ] Protect main with branch protection. Agents open PRs. Humans merge.
- [ ] Turn on secrets scanning (gitleaks, GitHub secret scanning, or equivalent) on every repo the agent touches.
- [ ] Keep CLAUDE.md short. Put hard never-rules in hooks and deny rules, not in wishful prompt text.
- [ ] Run the 60-second AGENTS.md check: Did you write it? Does it ask for network, email, or secrets? Does it change the agent's goal? If any answer is wrong, quarantine the file before chat.
- [ ] Review .mcp.json and installed MCP servers before the first session in a cloned repo.
- [ ] Use a feature branch and a clean git status before you delegate. Commit often so you can roll back.
- [ ] After the agent finishes, read the diff. Do not merge on trust alone.

13 items

CHECKLIST

Logging, monitoring, and kill switch

If you cannot see it, you cannot stop it. Watch for drift as well as break-ins.

- [ ] Log every tool call: time, agent id, user, tool name, target, approval id, result status.
- [ ] Redact secrets and PII from those logs before storage.
- [ ] Alert on spike in tool calls, repeated guardrail tripwires, denied high-risk actions, or spend over budget.
- [ ] Monitor agent inputs and outputs for data drift: gradual shift often means normal drift; abrupt change can signal poisoning or compromise (CSI AI Data Security).
- [ ] In healthcare and other high-stakes uses, prefer continuous monitoring of model performance plus direct input checks before patient impact.
- [ ] On Bedrock, enable tracing for agents where you need step-level visibility.
- [ ] On Codex, OTel is off by default. Enable it only to collectors you control, with prompts redacted.
- [ ] Set timeouts and max turns on agentic workflows so a stuck loop dies on its own.
- [ ] Maintain a written kill switch: who revokes keys, disables the agent alias, and rotates secrets.
- [ ] Test the kill switch once. A plan you never rehearse will fail under stress.
- [ ] Keep an incident channel and an on-call owner for production agents.
- [ ] Retain audit logs long enough for your industry needs, then delete them with a secure sanitization method.

12 items

CHECKLIST

Vendor and supply-chain risk

Your agent is only as safe as the tools, models, and datasets you bolt on.

- [ ] Inventory models, MCP servers, plugins, skills, datasets, and third-party tools the agent can call.
- [ ] Prefer signed, version-pinned components. Avoid latest tags in production.
- [ ] Treat web-scale and third-party training or RAG data as untrusted until verified. Watch for split-view and frontrunning poisoning on expired or crowd-sourced sources (CSI AI Data Security).
- [ ] Before ingest, verify dataset hashes or signatures. Discard anything that fails the check.
- [ ] Ask foundation-model and dataset vendors for assurances on training-data sources and known-compromise filtering. Prefer certified or attested suppliers when stakes are high.
- [ ] Sanitize and deduplicate third-party data. Validate metadata before it shapes an agent or model.
- [ ] Review changelogs before you upgrade model IDs or agent SDKs.
- [ ] Disable unused MCP servers and browser tools.
- [ ] Do not install coding-agent plugins from unknown publishers into prod machines.
- [ ] Watch for instruction files that arrive via dependency updates (ASI04 supply chain).
- [ ] Confirm data residency and subprocessors in vendor contracts before you send customer data.
- [ ] Have an exit plan: export prompts, tools, and logs if you leave a vendor.
- [ ] Re-run adversarial tests after any prompt, tool, memory, dataset, or provider change.

13 items

CHECKLIST

Before you hand the AI guy every login

For small-business owners hiring a contractor or agency. Print this page.

- [ ] Write down which systems the person may touch. Everything else stays off limits.
- [ ] Create separate logins for the contractor. Do not share your personal owner password.
- [ ] Turn on MFA everywhere before you share access.
- [ ] Prefer time-limited invites and role-based access over shared admin accounts.
- [ ] Never send production passwords over SMS or plain email. Use a password manager share.
- [ ] Do not give the contractor your bank, payroll, or EHR admin until there is a signed agreement and a need.
- [ ] Ask how they sandbox AI agents, where transcripts go, and whether customer data leaves their laptop.
- [ ] Require that AI tools used on your data are covered by a contract you accept (including a BAA if PHI is in scope).
- [ ] Keep an access list. Remove the contractor the day the project ends. Rotate anything they saw.
- [ ] Demand a short written handoff: what changed, which keys exist, and how to turn agents off.
- [ ] If they refuse least privilege or logging, pick someone else.

11 items

CSI CALLOUT

AI data hygiene (NSA/CISA/FBI CSI)

Plain-English takeaways from the May 2025 joint CSI AI Data Security sheet. Use with checklists (d), (f), and (g).

- [ ] Track where AI data came from and who changed it (provenance).
- [ ] Hash or sign datasets in storage and transit. Fail closed on mismatch.
- [ ] Classify data; keep outputs at the same sensitivity as inputs.
- [ ] Encrypt at rest and in transit. Mask PII before casual agent use.
- [ ] Watch three risk areas: data supply chain, poisoned data, and data drift (gradual vs abrupt).
- [ ] Delete retired AI media with secure sanitization (NIST SP 800-88 class).
- [ ] Reassess risks on a schedule with NIST RMF / AI RMF style reviews.

Source: NSA/CISA/FBI joint CSI AI Data Security (May 2025 Ver. 1.0), TLP:CLEAR.

INCIDENT

Incident quick-response card

Keep this page near your runbook. Speed beats perfection in the first 15 minutes.

1. Stop the agent. Disable the alias, pause the workflow, or shut the session.
2. Revoke credentials. Rotate API keys, IAM keys, OAuth tokens, and webhook secrets the agent could use.
3. Preserve evidence. Export logs, traces, prompts, and tool calls before you wipe anything.
4. Assess blast radius. What data was read? What was sent, posted, charged, or deleted?
5. Notify owners. Business owner, IT, and (if PHI/payment data) your privacy or compliance contact.
6. Block the path. Patch the prompt injection, remove the bad MCP/server, tighten deny rules, add the missing approval.
7. Reset trust. Re-issue least-privilege credentials. Do not reuse the compromised key.
8. Write the post-incident note: timeline, impact, fix, and the checklist item that would have caught it.

Contacts

Business owner: _____

IT / MSP: _____

Privacy / counsel: _____

Cloud account owner: _____

RISK TIERS

Risk-tier worksheet

Circle the highest tier any tool can reach. Apply that tier's required controls before go-live.

Tier: Low

Examples. Search docs, read public web, summarize a file the user attached, draft text for human edit.

Required controls. Scoped identity; input filters; logging; spend cap.

Tier: Medium

Examples. Edit files in a feature branch, create a ticket, write to a staging database, send an internal draft email to yourself.

Required controls. All Low controls + sandbox or non-prod account; output validation; allowlists; human review before merge or external send.

Tier: High

Examples. Send external email, post publicly, change IAM, deploy to prod, delete or export customer data, move money, touch PHI.

Required controls. All Medium controls + explicit human approval per action; return-of-control or needs_approval; separate prod credentials; kill switch tested; written owner.

Sign-off

Agent name / use case: _____

Highest tier: _____

Owner: _____

Date: _____

Kill-switch tested (Y/N): _____

REFERENCES

Sources

Primary sources fetched for this kit. Open each URL for the current vendor wording.

Amazon Bedrock Guardrails: https://docs.aws.amazon.com/bedrock/latest/userguide/guardrails.html

Bedrock sensitive information filters: https://docs.aws.amazon.com/bedrock/latest/userguide/guardrails-sensitive-filters.html

Bedrock denied topics: https://docs.aws.amazon.com/bedrock/latest/userguide/guardrails-denied-topics.html

Bedrock Agents user confirmation: https://docs.aws.amazon.com/bedrock/latest/userguide/agents-userconfirmation.html

Bedrock Agents return of control: https://docs.aws.amazon.com/bedrock/latest/userguide/agents-returncontrol.html

Bedrock Agents service role / IAM: https://docs.aws.amazon.com/bedrock/latest/userguide/agents-permissions.html

AWS Well-Architected Generative AI Lens (PDF): https://docs.aws.amazon.com/pdfs/wellarchitected/latest/generative-ai-lens/generative-ai-lens.pdf

Claude Code permissions: https://code.claude.com/docs/en/permissions

Claude Code permission modes: https://code.claude.com/docs/en/permission-modes

Claude Code security: https://code.claude.com/docs/en/security

Claude Code best practices: https://code.claude.com/docs/en/best-practices

OpenAI A Practical Guide to Building Agents (PDF): https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf

OpenAI Agents SDK guardrails: https://openai.github.io/openai-agents-python/guardrails/

OpenAI guardrails and human review: https://developers.openai.com/api/docs/guides/agents/guardrails-approvals

Codex agent approvals and security: https://learn.chatgpt.com/docs/agent-approvals-security

OWASP AI Agent Security Cheat Sheet: https://cheatsheetseries.owasp.org/cheatsheets/AI_Agent_Security_Cheat_Sheet.html

OWASP Top 10 for Agentic Applications (announcement): https://genai.owasp.org/2025/12/09/owasp-top-10-for-agentic-applications-the-benchmark-for-agentic-security-in-the-age-of-autonomous-ai/

OWASP Top 10 for Agentic Applications (download): https://genai.owasp.org/download/52117/

AGENTS.md and agent goal hijack (Prompt Security): https://prompt.security/blog/when-your-repo-starts-talking-agents-md-and-agent-goal-hijack-in-vs-code-chat

NIST AI 600-1 Generative AI Profile (PDF): https://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.600-1.pdf

NSA/CISA/FBI CSI: AI Data Security (May 2025 Ver. 1.0): https://media.defense.gov/2025/May/22/2003720601/-1/-1/0/CSI_AI_DATA_SECURITY.PDF

NSA/CISA/FBI CSI: Deploying AI Systems Securely (Apr 2024): https://media.defense.gov/2024/Apr/15/2003439257/-1/-1/0/CSI-DEPLOYING-AI-SYSTEMS-SECURELY.PDF

Optional source not fetched this session: Google SAIF. Direct curl to media.defense.gov returned 403; CSI text confirmed via WebFetch.

Built by Keith Motte · Topofmind.AI · Not legal advice · For SoCal local small businesses and the TopofmindAI Builders community. @kman62 · Topofmind.AI