

The Offerings Atlas: mapping what 166 AI platforms cover, and what they don't

M Walker Built 6 to 10 October 2026, release 1 on 14 October 2026

I built a knowledge base of AI agent security controls that agents research, verify and keep rechecking as vendors ship. It covers 166 platforms and tools, and its real product is the gaps: what each platform does not cover, labelled by how well that has been tested.

◇ At a glance

The atlas holds 7,304 researched findings across 44 controls and 166 platforms and tools, backed by 4,205 sources that are each tiered by who published them. 33 cells are agent verified, each with the stored page, its hash and a quote a pure function found on that page. It runs on my own machine through a local gateway with two model families, Claude and GPT, and no API keys. 250 automated tests pass.

It is a portfolio proof of concept. It is not a production service, a certification or advice about any vendor's current product.

◇ What changed on 14 October

A client mostly wants to know what a platform does not do, because that is where the work sits. So I tested the gap calls. Research agents took a sample of 30 cells blind: each saw only the platform and the control, never the atlas's answer, and searched the vendor's own documentation. Of the 24 cells the atlas called not addressed, the agents found a documented feature in seven and could not settle one more. About a third of the gap calls did not hold.

The misses were nearly always a recent release or a feature a thin first search missed. A Docker sandbox in LiteLLM's harness, sandbox egress allow lists in LangSmith and workflow version history in Kibana 9.5 all turned up. The agents had read a median of two pages a cell. The same method found the feature in all six cells known to be supported, so a search that comes back empty does carry weight when it is done properly. Release 1 is built on that result.

- **Gap labels.** Every in-scope gap is now confirmed, likely, unchecked or disputed. Confirmed means two searches to the standard found nothing, from two model families or from two runs at least 30 days apart. Likely means one did. Unchecked is a researcher's claim nobody has tested yet. Disputed means a check found a feature with a quote that passed, so the cell goes back to research. The label comes from the checks on record, never from a model's confidence, which in the sample tracked how many pages it had read.
- **The search standard.** A check only counts if it fetched and hashed at least three pages from the vendor or another first-hand source: the documentation index, a search of the documentation site with the query recorded, 12 months of release notes, and the API reference or repository where there is one. Gated documentation is recorded as insufficient evidence, with the reason.

- **Rechecks on release notes.** Each platform's release notes sit on a watch list. A change sends its likely and confirmed gaps back for checking, so a gap does not outlive the release that closed it. Without a change, they are rechecked every 90 days.
- **Gaps agents may use.** A confirmed or likely gap can be retrieved by an agent on a new basis, absence checked, with its label shown. It is never marked verified, because nothing can prove an absence with a quote. Anything a client relies on needs confirmed.
- **Coverage.** An audit compared the columns with the major vendors in each group and found 14 missing. All 14 were researched and added, among them Red Hat OpenShift AI, Chrome Enterprise Premium, VMware Private AI Foundation, NICE CXone Mpower and Rubrik Agent Cloud. Their not-addressed verdicts cite the pages that were checked.
- **Two model families.** The judge now runs Claude Haiku and GPT-5.5 through the gateway. Run against the labelled test set with both, it accepted no altered claim and held back one true claim for a closer look.

The first group to go end to end was AI security: Cisco AI Defense, Palo Alto Networks Prisma AIRS and Wiz. Of 70 claims checked, 13 were verified, 30 were only partly supported and went back to be split, and 21 had too little evidence, mostly on Cisco's pages. Across the atlas, 7 gaps are now confirmed, 2 likely and 7 disputed. 1,538 in-scope gaps are still unchecked.

Everything else is queued rather than listed. Every priority cell that is not yet verified or absence checked is a job: 3,942 first claim checks and 1,517 first absence checks, about 8,556 judge calls in all. At the daily cap of 400 calls the first pass clears around 5 November.

◇ What it demonstrates

- **Agent design.** Four agent roles with separate runs and narrow permissions, none of which checks its own work.
- **Knowledge engineering.** One controls matrix projected into an OKF v0.2 tree, with frontmatter, chunks and an eligibility rule for agents.
- **Provenance.** Every verified claim rests on a stored page, a hash and a quote that can be checked again without trusting the agent.
- **Evaluation.** A labelled test set with deliberately altered claims, run against the real judge before it was allowed to write.
- **Governance.** Facts run with no human gate, while advice for an enterprise goes through a pull request with named approver roles.
- **Cost engineering.** Change detection by section hash, so only changed pages go back to a model.

◇ Why I built it

Clients ask me the same questions about agents. How does agent-to-agent authorisation work, and how do we stop one agent acting with another agent's permissions? Which gateway do we need in front of models and MCP tools? Can prompts, outputs and memory stay in Australia? Who approved that action, and can we prove it?

The answers are spread across hundreds of vendor documentation pages, and they change every few weeks. A model asked directly will answer fluently and often be out of date. I wanted a knowledge base that agents build and keep current themselves, where every fact traces to the exact passage it came from, and where it is clear what a person still has to decide.

◇ The matrix

The atlas is one controls matrix. The 44 controls are rows, grouped into identity, authorisation, runtime, mediation and policy enforcement, observability, agentic MLSecOps and governance. Each platform or tool is a column, grouped by what it does: frontier AI providers, cloud platforms, coding agents, identity and secrets, AI security, AI governance, AI and MCP gateways, Australian sovereign hosting and 13 more.

Verdict	Cells	Meaning
Supported	810	The vendor's own documentation describes a native feature that meets the control
Partial	3,357	A native feature covers part of it, or needs configuration, add-ons or a higher tier
Not addressed	3,033	Checked, and the product does not offer it, either because it sits outside its category or as a gap that carries a label
Unresolved	104	The documentation does not settle it either way

A verdict describes documented capability within the researched scope. It is not a deployment test, and the atlas says so wherever a verdict appears.

The atlas has a dashboard of its own, and the two screens below are from the full version, not the public sample. Both were taken on 10 October, before release 1, so the totals across the top are a little behind the figures above. The first is the matrix scrolled across the AI security, AI usage control and data security groups. The second sets three tools from outside the sample side by side. Each cell carries its verdict, its answer, its gap and the date it was checked, and on that day every one of them was labelled unverified because no evaluator had run on it. All three went through the evaluator on 14 October.

CONTROLS MATRIX

Controls matrix

Rows are the 44 controls. Columns are the 151 platforms and tools in the coverage list (decision 0006), grouped by what they do. Select a cell for the finding and its sources.

Add control

151

Platforms and tools

44

Controls

6644

Researched cells

725

Supported

3007

Partial

2838

Not addressed

74

Unresolved

20

Agent verified

Column group

All column groups

Priority

P1 and P2

Control category

All categories

Highlight

All verdicts

Search

Platform, tool or control

Yes Supported Part Partial No Not addressed ? Unresolved - Not researched Part Agent verified, with stored provenance Other cells are researched with cited sources and owner accepted.

Control	AI SECURITY																		AI USAGE CONTROL (SASE, BROWSER, DLP)			DATA SECURITY FOR AI				
	Cisco AI Defense	Palo Alto Networks Prisma AIRS	Wiz	Akamai Firewall for AI and Workforce Protector	Lukera	F5 AI Guardrails and AI Red Team	Fortinet Security for AI (FortiAI Gate and Virtue AI)	HiddenLayer	Noma Security	Pillar Security	Prompt Security (SentinelOne)	Proofpoint AI Security	Snyk Ego and Agentlic Development Security	Straker	Tenable One AI Exposure	Trend Vision One AI security	Zenith	Nextops One AI Security	Zscaler AI Security (with SPLX)	Cato AI Security	Island Enterprise Browser	Microsoft Purview (DSPM for AI, Compliance Manager)	BigID AI governance and agentic access control	Cyera (with Oasis)	Immuta Agentic Data Access	
Identity																										
Agent ID	No	?	Part	No	No	No	No	No	Part	No	Part	No	No	No	No	No	Part	No	?	Part	Part	No	No	Part	Part	
Credential Brokerage	No	Part	No	No	No	Part	Part	No	No	No	No	No	No	No	No	No	No	Part	No	No	Part	No	No	Part	Part	
Delegation (OBO)	?	No	No	No	No	No	No	No	No	No	No	Part	No	No	No	No	Part	No	No	No	Part	No	Part	Part	Part	
A2A	No	?	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	?	No	No	No	No	No	No	
Authorisation																										
Agent Local Scopes	No	Part	No	Part	No	?	No	Part	Part	Part	Part	Part	Part	Part	No	No	Part	No	Part	Part	Part	No	No	Part	No	
Agent Remote Scopes	Part	Part	No	Part	Part	Part	Part	Part	Yes	Part	Part	Yes	Part	Part	Part	Part	Yes	Part	Part	Part	Yes	Part	Part	Yes	Part	
Human Approval	No	No	No	No	No	No	No	No	Part	Part	No	No	No	No	No	?	No	No	No	No	Part	No	No	Part	No	
Agent Provisioning	No	No	No	No	No	No	No	No	Part	No	No	No	No	No	Part	No	Part	No	No	No	Part	No	No	Part	Part	
Runtime																										
Agent Orchestration	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	Part	No	No	No	No	

44 controls shown for 3 platforms and tools.

Control	Cisco AI Defense AI security	Palo Alto Networks Prisma AIRS AI security	CrowdStrike Falcon (AIDR, Next-Gen SIEM, Identity) Security operations (SIEM, XDR)
Identity			
Agent ID	Not addressed Open AI Defense does not issue or manage agent or workload identities. Checked 2026-10-09 unverified	Unresolved Open The docs navigation has an AI Agent Identity section, but the pages opened cover SaaS agent onboarding and the MCP server, not identity issuance. Gap: No documented agent identity issuance, owner linkage or release binding found. Checked 2026-10-09 unverified	Partial Open The Agentic Identity Provider and Continuous Identity for AI Agents releases describe a cryptographically verifiable identity for each agent based on SPIFFE, registered automatically when Falcon Guardian discovers the agent, with actions tied to an accountable agent. Gap: Described only in press releases with forward-looking statements. Checked 2026-10-09 unverified
Credential Brokerage	Not addressed Open AI Defense does not deliver or rotate credentials for agents. Checked 2026-10-09 unverified	Partial Open AI Gateway holds provider integrations behind workspace virtual keys, so applications send a gateway API key and virtual key rather than provider secrets. Gap: Rotation, expiry and revocation of gateway keys are not described. Checked 2026-10-09 unverified	Partial Open The Agentic Identity Provider release says agents get no standing credentials and receive short-lived tokens scoped to the minimum access for the minimum time per task. Gap: No documentation of token formats, rotation, revocation API or how secrets are kept out of model context. Checked 2026-10-09 unverified



◇ Four agents, each with one job

The roles come from a decision record, and each runs separately with no shared context.

Role	Does	Never does
Researcher	Reads vendor documentation and writes one finding per cell, with citations	Marks anything verified
Evaluator	Fetches each cited page, stores it by hash, has a model pick a verbatim passage, then checks the passage is really on the page	Judges a claim its own run drafted
Freshness	Hashes the sections a claim depends on and queues rechecks when they change	Rewrites a fact
Curator	Writes results through the store, runs the checks and commits with the run ID	Writes past a failed check

Every model call goes through a local gateway I built for this, which runs Claude Code and Codex headless on my own subscriptions and returns JSON that matches a schema. That gives the atlas two model families, Claude and GPT, so a second opinion never comes from the same model. The gateway is a separate project.

◇ Checkable proof

I did not want verification to mean an agent saying a claim was true. A verified cell in the atlas carries what anyone needs to check it again:

- the raw page, stored in a private evidence repository under its SHA-256
- the hash of the normalised text, the fetch time and the HTTP status
- a passage of 40 words or fewer, copied verbatim from that page
- the result of a pure-function check that the passage is an exact substring of the page text

The model decides whether the page supports the claim. It cannot invent the evidence, because a quote that is not on the page fails the check and the cell stays unverified. One command reopens every stored page and confirms every passage again, with no model involved. On the 10 October showcase run it reproduced all 78 stored passages.

Evaluator provenance

Agent verified

SUPPORTEDQUOTE CHECK PASSTIER 1

| 'auth_data' | 'object' | OAuth2 'client_id' and 'client_secret'. Required when the MCP server uses OAuth. |

<https://docs.mistral.ai/studio/connectors/management>

Fetches 2026-10-10T04:34:07Z | HTTP 200 | sha256 96496d35934151dd | claude-cli-sonnet via llm-gateway | run eval-2026-10-10-showcase

SUPPORTEDQUOTE CHECK PASSTIER 1

| Nothing long-lived is stored in the workload's environment.

<https://docs.mistral.ai/admin/identity-access/workload-identity>

Fetches 2026-10-10T04:36:02Z | HTTP 200 | sha256 ca6abdb88f3a69d1 | claude-cli-sonnet via llm-gateway | run eval-2026-10-10-showcase

SUPPORTEDQUOTE CHECK PASSTIER 1

| rotate a key by creating a new key and deleting the old one;

<https://docs.mistral.ai/admin/identity-access/api-keys>

Fetches 2026-10-10T04:34:09Z | HTTP 200 | sha256 34f4afa465e40a82 | claude-cli-sonnet via llm-gateway | run eval-2026-10-10-showcase

Judge: Connector auth_data, static headers and per-user credentials, workload identity federation with no stored secret, and API key expiry, policy, rotation and deletion are all stated. Tool calling through connectors is only implied via direct tool call links.

The cell above is Mistral AI's credential brokerage. The judge found three passages across three of Mistral's own pages, one each for connector credentials, workload identity federation and API key rotation. The rejected cells are as useful. For Mistral's tool gateway the judge copied a whole permissions table, 41 words, so the quote check failed. It also found that no page said API keys were scoped to connectors. The cell stayed unverified with both reasons recorded, ready for a researcher to split the claim.

◇ Testing the judge before trusting it

Before the evaluator could write anything it had to pass a labelled test set of 67 cases. Of those, 27 are deliberately altered claims: wrong features, wrong numbers, outdated statements, quotes that are not on the page, and a page carrying a prompt injection. The pass mark was no altered claim accepted and no more than one true claim in ten rejected.

Judge	Altered claims accepted	True claims rejected
Scripted replies	0 of 27	0 of 40
Haiku, after the fixture labels were corrected	0 of 22	1 of 28
Sonnet	0 of 22	2 of 28

Live judges see 22 of the altered cases, since five alter the scripted judge reply rather than the claim. Both models disagreed with the same five labels on their first runs, and each time the fixture was wrong, not the judge. A judge rigged to agree with everything shows what the hash and quote checks catch on their own: it would have let 20 of the 27 altered claims through. The deterministic checks stop invented quotes and weak sources. Wrong features and outdated claims still rest on the judge.

On 14 October the set ran again on this machine with the second family in place. With Claude Haiku and GPT-5.5 judging together, 60 cases came out exactly as labelled, no altered claim was accepted and one true claim was held back for a closer look.

◇ The questionnaire

This is the part the rest of the atlas exists for. A client gives its maturity stage, what it runs and what worries it, in its own words. The example is a synthetic financial services firm piloting coding and support agents, running Claude Code on AWS with Okta, Wiz and Microsoft Purview.

1. **Route.** A small model maps each concern to the controls it is about, and to the tool groups where an answer might sit. A question about which gateway to use goes to the model and tool gateway controls and to the AI and MCP gateways group.
2. **Retrieve.** Records come from the OKF tree. Where nothing in the client's stack supports a control, the best-supported alternatives from the routed groups are added. Kong AI Gateway came up for agent-to-agent mediation.
3. **Answer.** A second model answers each concern from those records alone. Every claim must name a record and a source URL, and both are checked against the records it was given.
4. **Frame.** The client's maturity stage picks the engagement package and the offerings that fit it.

The run took five model calls, about 45,000 tokens in and 9,600 out. Of 38 citations, 37 passed the check. The one that failed, a claim about Okta's gateway that came back with no source URL, is marked as failed in the answer the client sees. The answers are direct about gaps. The records do not confirm that prompts, outputs and memory can stay in Australia end to end. Claude Code's auto memory has no documented retention limit. Wiz and Okta do not cover residency. Each answer ends with what the knowledge base does not settle, which is where an engagement starts.

◇ An OKF knowledge tree

The curator projects the atlas into an Open Knowledge Format v0.2 tree of 7,682 concepts, built to sit beside my business knowledge tree. Each matrix cell becomes a concept with YAML frontmatter and Markdown chunks:

```
---
type: "Platform capability"
title: "Anthropic: Human Approval"
classification: "C1"
verification: "unverified"
tags: ["frontier-ai", "authorisation", "supported"]
sources:
  - id: "an-permissions"
    resource: "https://platform.claude.com/docs/en/managed-agents/permission-policies"
    locator: "always_ask, auto and tool confirmation"
atlas:
  agent_eligible: true
  eligibility_basis: "owner-accepted"
```

```
---
<!-- chunk: answer -->
## Answer
...
```

Tags come from the column group, the control category and the verdict. Sources keep their locators and tiers. The eligibility rule decides what an agent may retrieve. In an enterprise tree only verified records would qualify. For this proof of concept I also accepted the researched cells myself, and those are labelled as owner accepted, never as verified. A cell the evaluator checked and could not verify drops out of the tree, because an evaluator verdict replaces my acceptance.

Release 1 added a third basis, absence checked, for gaps that have survived a search to the standard. Of the 7,682 concepts, 6,507 are open to agents: 33 agent verified, 9 absence checked and 6,465 owner accepted. My acceptance covers research up to 10 October, so the 616 cells added on 14 October wait for their checks before an agent can use them.



◇ Keeping it fresh, and where a person comes back in

The freshness agent watches only the sections of a page that a claim depends on, so a footer edit raises nothing. In a scripted demonstration, the Amazon Bedrock model card for Claude Sonnet 5.5 gains an Australian Geo inference route. The agent detects the changed section and finds the two findings that depend on it. It marks them stale and queues evaluator jobs, with no model call at any step.

For facts in this atlas there is no human gate. The rules are strict enough that an unchecked claim is still labelled as one, and a wrong claim is caught by the next evaluator run. Enterprise knowledge is different. The same change becomes a pull request with the original wording, the evidence hash, the verbatim passage and proposed new wording, routed through a knowledge owner, a security reviewer and a

publisher. The pull request also separates the fact from the judgement that stays with a person: whether client guidance that recommended other models for Australian processing should now include Sonnet 5.5. The approval chain is a labelled simulation with synthetic roles.

◇ What it costs

The gateway logs tokens and a notional price for every call. Calls on my subscriptions have no marginal cost, so the price is the tool's own estimate at list rates.

Work	Model calls	Notional cost
One Sonnet judge call on a real cell	1	about USD 0.045
The 41-cell showcase run	41	about USD 1.85
The client questionnaire	5	about USD 0.30
Verifying every supported and partial cell once	about 4,567	about USD 206 with Sonnet

The release 1 runs measured the real load: one judge call for each claim or gap check per model family, two for a high-risk claim. After the first pass, freshness and the release-note watch send only changed pages back to the judge, so keeping the atlas current costs a small share of building it.

◇ Lessons and where the pattern goes next

- **Separate the proof from the judgement.** A hash and a substring check are cheap and certain. The model's job shrinks to the part that needs judgement.
- **Test the judge against the failures you fear.** The gullible-judge run showed exactly which failures the deterministic checks cover and which still need a model.
- **Label every state honestly.** Researched, owner accepted, verified and withheld are different things, and a reader should never have to guess which one they are looking at.
- **Keep advice with a person.** Agents can keep facts current. Whether a fact changes a recommendation is a decision with an owner.

The same pattern fits any knowledge an enterprise's agents act on: control libraries, supplier assurance, policy registers, product catalogues. Each needs the same parts: tiered sources, checkable proof, change detection and a clear line between facts and decisions.

◇ What's next

- **Work the queue.** About 8,500 checks are queued across the priority platforms, at a daily cap, with the first pass due in early November. Researcher tasks run at five a day, so splitting claims and deeper research on gaps take longer than the judge.
- **Grow Ask the Atlas.** The public sample widens so the questionnaire covers more of the stack people actually run, with Snowflake and Databricks first.
- **Client grade.** Signed releases, access-controlled delivery and confirmed gaps only for anything a client relies on.

◇ Limits

- This is a portfolio proof of concept, not a production service, a certification or advice about any vendor's current product.
- Verdicts describe documented capability, not tested tenant behaviour.
- 33 cells carry agent proof and 16 gaps carry absence checks. Most cells still wait in the queue, and most gaps are unchecked, so an unchecked gap is a lead to test rather than a finding.
- The research dates from 7, 9 and 14 October 2026, and vendor products change.
- The client is synthetic, and no client or employer material is in the knowledge base.

◇ Sources

- [Open Knowledge Format README](#), Google Cloud
- [Workload identity and connector management](#), Mistral AI documentation, fetched 10 October 2026
- [Claude Sonnet 5.5 model card](#), Amazon Bedrock User Guide

Figures come from the project's own repository, evaluator runs and gateway proof log as of 14 October 2026.