



THE DIGITAL INFRASTRUCTURE CONTINUUM

AdapterCloud

One control plane that turns every cloud account, cluster, data centre, edge site and SaaS tenant you run into a single governable fabric — inventory, dependencies, policy and cost on one record.

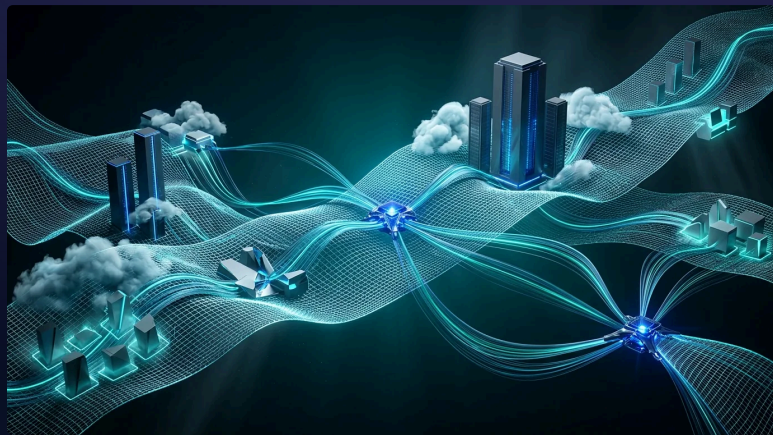
UNIFIED INVENTORY

DEPENDENCY TOPOLOGY

POLICY AND DRIFT

FINOPS

AI-NATIVE



Pre-launch. Launching 2026, waitlist open. Every capability in this document is marked shipped, partial or planned.

01 The short version

1 OF 3

Most organisations cannot answer four simple questions about their own infrastructure: what do we run, what depends on what, what does it cost, and is any of it against the rules. AdapterCloud makes all four a query.

Past a single cloud account, infrastructure becomes a set of islands. One provider console here, another there, a container cluster the platform team babysits, a rack in a data centre nobody quite owns, devices at branch sites, network gear behind its own management screen, and a long tail of SaaS tools each with its own admin panel. The inventory is a spreadsheet that is wrong the day it is saved, and nobody trusts it enough to make a decision on it.



One record per resource

AdapterCloud sits underneath all of that. You define an adapter — a reusable connector for one class of system — and a connection for each real, authenticated target. Discovery then fills a single inventory with what actually exists, and you record the dependencies between those things as a topology, so the estate stops being a list and becomes a graph. Policies, violation records, drift records and cost records all attach to those same entries.

That is the whole idea: one record per resource. When cost, dependency, policy and drift all point at the same entry, there is nothing to reconcile at month end, no export to merge, and no argument about which system is right. The product is honest about its stage — the control plane and the API are live, discovery runs against cloud accounts and container clusters today, and everything still being built is labelled as such throughout this document.

THE PROMISE

Ask what you run, what depends on it, what it costs and whether it is in policy — and get one true answer, from one system.

10

Classes of system

Public cloud, private cloud, on-premises, edge, connected devices, container clusters, network, data systems, SaaS and fully custom are peers in one model — not extensions bolted onto a cloud tool.

One

Record per resource

Inventory, dependency edges, policies, violations, drift records and spend all reference the same entry. No export, no matching, no monthly reconciliation.

Day 1

Tenant-isolated and role-checked

Every read is scoped to your workspace and every write proves the record belongs to you before it changes. Isolation is architectural, not a configuration you have to get right.

Free

Tier with no card

A forever-free plan with the full product surface at activation-sized quotas, so you can prove value before anyone signs anything.

Platform and infrastructure leads

You are accountable for an estate spread across more than one cloud, a data centre, a cluster fleet and a pile of SaaS. You need one inventory and one dependency picture that is current, not a wiki page written by someone who left.

Cloud and site reliability engineers

You want configuration drift surfaced as a tracked item before it becomes an incident, and you want to answer “what breaks if this goes away” from recorded dependencies rather than from memory.

FinOps and cloud-cost teams

You need spend attributed to the same resources the platform team operates, so a disputed line item is a lookup rather than a ticket, a wait and a guess.

Security and governance officers

You want guardrail, tagging, security, cost, naming and compliance rules to live in the system as data, with violations that have a real lifecycle and an audit trail rather than an email thread.

Managed-service providers

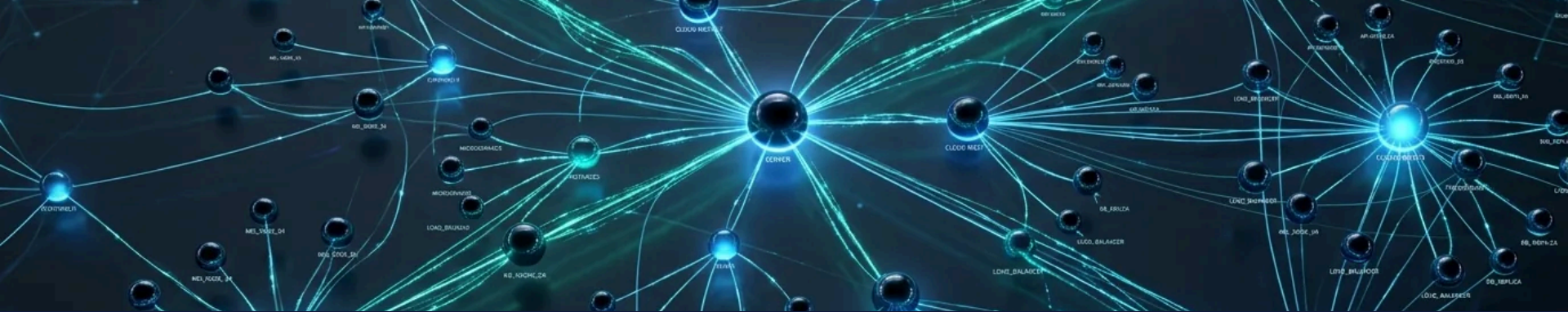
You run many client estates and need one operating model with per-client isolation that is architectural, so onboarding a new client is days rather than a fresh tool configuration.

Platform builders and integrators

You are building an internal developer platform and want a control-plane API you can query directly, plus the ability to bridge your own internal systems in with a custom adapter.

Organisations consolidating sprawl

You grew into fragmentation by acquisition or by accident, and you want one governable picture before the next audit, the next migration or the next surprise bill.



03 What breaks today

None of these are exotic failures. They are the ordinary, daily cost of running infrastructure that grew faster than the tooling around it — and every one of them is a symptom of the same root cause: there is no single record of what you actually run.

The inventory is wrong the day it is saved

A quarterly spreadsheet exercise cannot keep pace with an estate that changes hourly. Every decision built on it carries an error margin nobody can quantify, and the people who know it is wrong update it anyway.

- Discovery registers each resource against its own provider-native identifier, so re-running it updates the existing entry instead of creating a duplicate. The inventory converges towards the truth instead of sprawling away from it.

Nobody can trace what depends on what

“If I change this security group, what breaks?” gets a shrug and a suggestion to ask the team on the other end — which nobody can name. Every change is a risk and every incident is a surprise.

- Dependencies are recorded as typed edges between resources — depends-on, connects-to, contains, routes-to, replicates-to. Selecting a resource shows its direct dependencies and its full downstream set, computed by walking real recorded facts.

Policy lives in a wiki, enforcement lives in someone's head

“All resources must carry an owner tag” is a sentence in a document. The audit is a spreadsheet. The gap between what was agreed and what is running is discovered once a quarter, by hand.

- A policy is a first-class entry with a kind, an enforcement mode, a severity, a rule expression and a scope selector. It references the same resources you operate, so compliance becomes a property of the live estate.

Drift is discovered during the incident

A manual fix here, an out-of-band change there, and running state quietly diverges from the blueprint. The divergence is found at the worst possible moment, by the person paged at three in the morning.

- A drift record captures the exact setting that changed, the value expected, the value observed and a severity — then moves through detected, acknowledged, remediated or ignored, so divergence is a work item rather than a landmine.

Cost and inventory never agree

Finance sees a bill go up. The platform team has an inventory. The two are reconciled by hand, over days, every month — and a business unit disputing a charge cannot be answered without a ticket and a wait.

- Spend is ingested as period-bounded cost records attached to the same resource or connection the rest of the estate references, and rolled up by service across the workspace. There is nothing to reconcile because there is only one set of records.

A different console for every provider

Five consoles, three spreadsheets, two wikis and a set of scripts that break whenever an API changes. Answering a simple question means logging into everything and merging the results by hand.

- One workspace-scoped control plane holds inventory, topology, governance and cost for every connected environment, with the same navigation, the same permission model and the same audit trail across all of them.

Governance findings have no lifecycle

A finding is raised, mentioned once, and sometimes fixed. There is no queue, no status, no owner and no record of the deliberate exceptions — so the same finding reappears next quarter as if it were new.

- A violation carries a message, evidence and a severity, and moves through open, acknowledged, resolved or waived. Waiving demands a reason, so an accepted risk is always explainable later.

Every new estate needs a new tool configuration

For a service provider or a group with many business units, each new environment means fresh scripts, fresh dashboards and fresh processes. The tooling, not the team, becomes the ceiling on growth.

- One adapter and connection model covers every class of system, and every entry belongs to a workspace. Onboarding a new estate is a connection and a discovery run, under the same operating model as all the others.

AdapterCloud is deliberately built from a small number of concepts. Learn these nine and you know the product — everything else in the interface is a view over them.

FABRIC

Adapters

A reusable connector definition for one class of system, carrying a kind, a vendor, a version, declared capabilities, an optional configuration schema and a publication lifecycle from draft through published and deprecated to archived. Ten kinds ship, including connected-device fleets and network gear as peers of public cloud.

FABRIC

Connections

A configured, authenticated instance of an adapter pointed at one real target — an account, a cluster, an endpoint, a tenant. It records the sign-in method, a reference to a stored secret, a target reference, region, environment, resource count and a health state that moves through a validated lifecycle.

FABRIC

Resource inventory

Every discovered thing, in one list you can filter by connection, type or lifecycle state. Each entry carries a provider-native identifier, type, name, region, provider tags, free-form attributes, monthly cost, a last-seen timestamp and a state from discovered through reconciled, drifted, orphaned or deleted.

FABRIC

Topology

Typed dependency edges between resources, each with a kind, an optional weight, a direction flag and attributes. The map zones the estate by class of system, highlights edges touched by drift, and answers direct-dependency and downstream blast-radius questions by walking what has been recorded.

GOVERNANCE

Policies

Rules as data, across six kinds — guardrail, tagging, security, cost, compliance and naming — with three enforcement modes, a severity, a rule expression and a selector for the part of the estate they apply to. Policies are listed, filtered and evolved like any other record.

GOVERNANCE

Violations

A specific resource breaching a specific policy, recorded with a message, structured evidence, a severity and a detected timestamp. Its lifecycle is validated end to end — open, acknowledged, resolved or waived — and a waiver cannot be recorded without a reason.

GOVERNANCE

Drift records

One divergence between observed and desired state, naming the exact attribute path, the value expected, the value observed and a severity, with its own status and a timestamped remediation. Drift becomes a filterable queue rather than tribal knowledge.

OPERATIONS

Sync runs

One execution against a connection — discovery, reconcile, cost ingest, or all of it. Each run moves through queued, running and then succeeded, failed or cancelled, and records start and finish timing, any error, and counters for resources discovered and reconciled and for drift and violations produced.

FINOPS

Cost records

Period-bounded spend with an amount, a currency, a service, a period and a structured breakdown, attributed to a resource or a connection. Because it attaches to the operating record rather than to a separate billing model, cost is a view of the estate rather than a second dataset.

05 Put to work

These are illustrative scenarios written against what the product does, not case studies — AdapterCloud is pre-launch and has no adoption to quote. Each one names honestly where the capability is live and where it is still being built.



PLATFORM ENGINEER

Inherit an estate and get a real picture of it in a week

You define an adapter for each class of system and create a connection per real target. A discovery run populates the inventory against provider-native identifiers, so re-running it converges instead of duplicating. You then record dependency edges between the load balancer, the cluster and the database, and the estate stops being a list.

- One current, filterable inventory and a dependency graph you can defend in a change review — instead of a spreadsheet you already know is wrong.

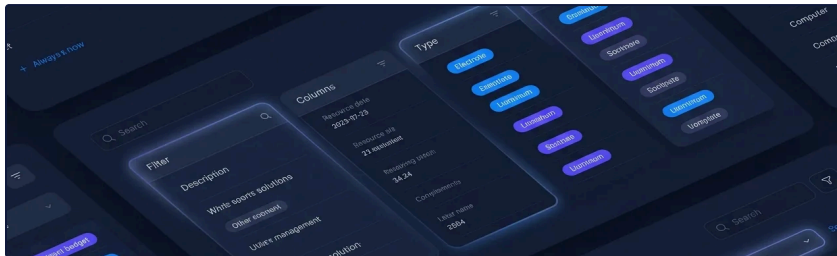


INTEGRATION ENGINEER

Bridge systems that were never designed to talk to each other

An adapter maps one class of system into a single shape — public cloud, private cloud, on-premises, edge, connected devices, container clusters, network, data systems, SaaS or fully custom. A connection then points that adapter at one authenticated target, with a sign-in method, a region, an environment and a health lifecycle. Credentials never enter the inventory: a connection holds only a reference to a secret in a vault.

- New systems join by adding an adapter and a connection, rather than by re-plumbing everything around them.

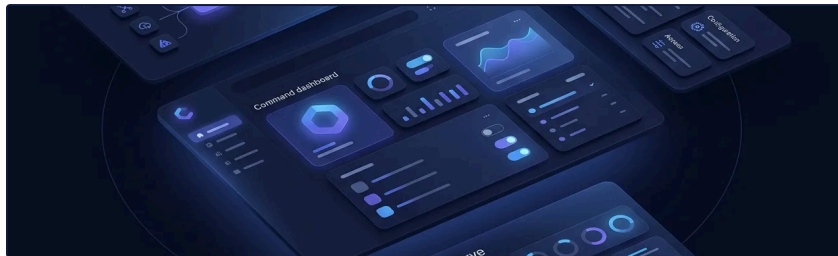


CLOUD ENGINEER

Answer “what do we actually run” without a week of archaeology

Every discovered resource carries a type, a name, a region, provider tags, attributes, a monthly cost, a last-seen timestamp and a lifecycle state. You filter the whole estate by connection, by resource type or by state — every drifted database, every orphaned load balancer, every resource in one region — and the same entries are the ones policy, drift and cost already reference.

- Estate questions become queries with a timestamp on the answer, not projects with a deadline.

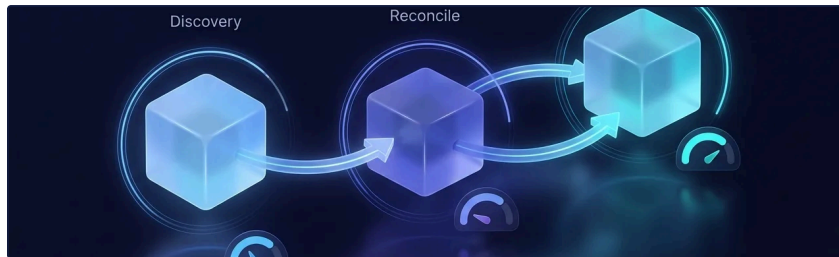


PLATFORM ENGINEER

Run every environment from one surface instead of six

Ten workspace-native pages hold the whole model: an estate overview with fabric counts, connection health, open violations, unremediated drift, recent runs and cost by service; then adapters, connections, resources, topology, policies, violations, drift, sync runs and a FinOps view. All of them call the same permissioned interface your automation calls.

- One navigation, one permission model and one audit trail across every connected environment.

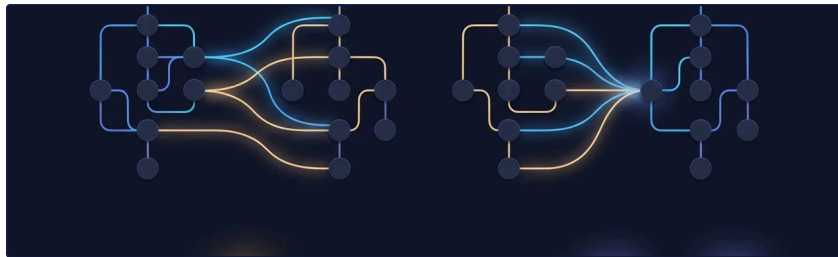


SITE RELIABILITY ENGINEER

Keep the inventory current by construction

A sync run is one execution against a connection with a kind: discovery, reconcile, cost ingest, or a full pass. It moves through a validated lifecycle and records when it started, when it finished, any error it hit, and counters for what it discovered, what it reconciled, and how much drift and how many violations it produced.

- “Did the inventory refresh, and what did it find?” is a query against an audited record rather than a guess.



SITE RELIABILITY ENGINEER

Catch configuration drift before it pages somebody

A drift record names the exact attribute path that changed, the value expected, the value observed and a severity. It carries its own status — detected, acknowledged, remediated or ignored — and remediation is timestamped for the audit trail. You filter drift by resource or by status and work it like a queue.

- Divergence becomes a visible, owned work item instead of an invisible risk you find the hard way.



SECURITY AND COMPLIANCE LEAD

Move governance out of the wiki and into the system

A policy is an entry with a kind — guardrail, tagging, security, cost, compliance or naming — an enforcement mode, a severity, a rule expression and a selector scoping it to part of the estate. Because it references the same resources you operate, the rule applies identically across every connected environment in the workspace.

- Compliance posture becomes queryable and continuous rather than a snapshot that is stale before the meeting ends.



GOVERNANCE LEAD

Work non-compliance as a queue with an audit trail

A violation records the message, structured evidence, a severity and a detected timestamp, then moves through a validated lifecycle: open, acknowledged, resolved or waived. Resolving stamps who and when. Waiving requires a reason, so an accepted risk is always explainable. You filter the queue by policy, resource or status.

- Every exception has a name and a reason attached, which is exactly what an auditor asks for.



FINOPS LEAD

Attribute spend to the things you actually operate

Cost records carry an amount, a currency, a service, a period and a structured breakdown, attached to the same resource or connection the inventory, topology, policies and drift records reference. A workspace-wide summary rolls spend up by service, optionally scoped to one connection, and records can be filtered by connection, resource or service.

- A disputed line item is a lookup on the estate map instead of a ticket, a wait and a monthly reconciliation project.

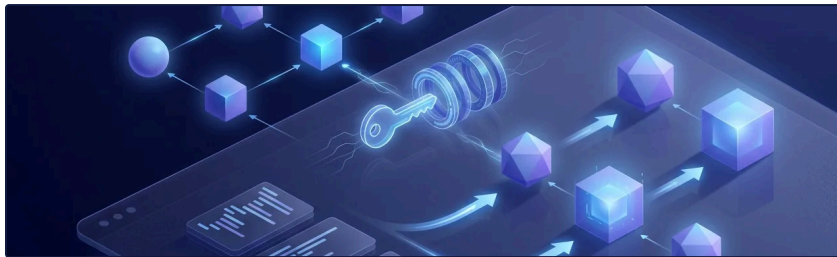


PLATFORM OWNER

Let many teams share one control plane safely

Every adapter, connection, resource, edge, policy, violation, drift record, run and cost record belongs to a workspace. Every operation carries a permission requirement checked before the request reaches the data. Isolation is enforced on writes too: an update only succeeds when exactly one record in the caller's own workspace matches, so a guessed identifier from another tenant is rejected rather than mutated.

- Multi-tenancy, permissions and audit are properties of the architecture rather than a configuration you have to keep getting right.

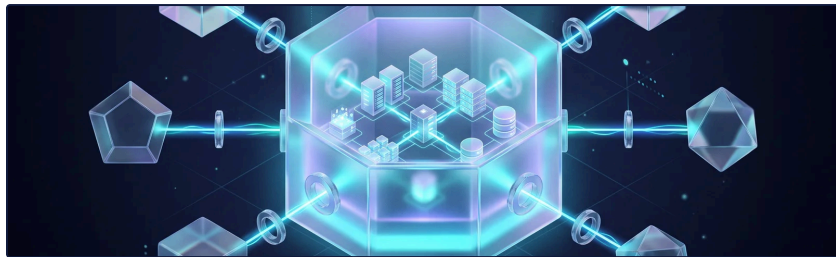


PLATFORM BUILDER

Automate the estate from your own pipelines

The whole control plane is one API, and the product's own screens call it. You register an application in the developer portal, mint a key scoped to exactly the permissions it needs with an expiry, and store the secret — shown once at creation. From there you list connections, register resources, add edges, record violations, start a run or pull a cost summary from scripts or a pipeline, through the same permission checks and the same audit trail as a click.

- Anything a person can do in the product, a pipeline can do too — with no side door and no separate automation surface to keep in step.

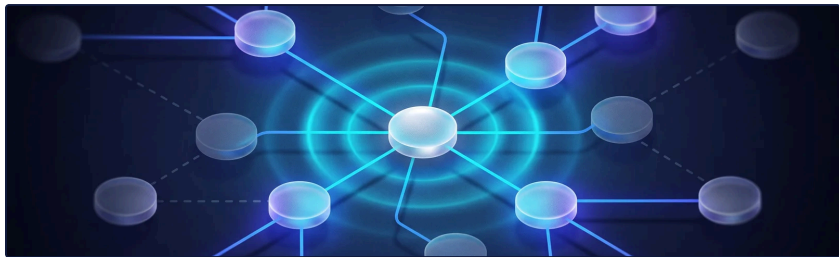


PLATFORM AND SRE ENGINEERS

Point your own AI client at the estate, signed in as you

AdapterCloud exposes 37 operations as tools over the Model Context Protocol, covering adapters, connections, inventory, topology, policies, violations, drift, runs and cost, plus four read-only estate snapshots so an agent can orient before acting, and two guided workflows: connection onboarding, and a governance sweep. The server signs in with your own identity through a device-code flow, so every call carries your permissions and your audit trail.

- Your agent works the estate with exactly your access — no shared service key to leak into a configuration file.

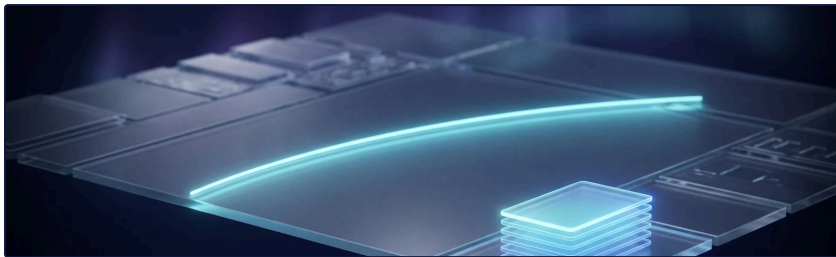


CLOUD AND SRE ENGINEER

Ask what breaks if this goes away, and get arithmetic

Dependency edges carry a kind, a direction, an optional bidirectional flag and a weight. The topology map builds forward adjacency from them and computes, for any node you select, its direct dependencies and its full transitive downstream set. There is no model and no confidence score involved — it counts paths that were actually recorded, and it is only as complete as what discovery and your own edges have captured.

- Change reviews get a number with a stated basis, and we say plainly that an unrecorded edge makes that number look smaller than reality.



PLATFORM OWNER

Plan for the sweep nobody has time to run

This one is designed and committed, not built. The planned loop checks connection health, looks for runs that failed or never finished and inventory gone stale as a result, ages the drift board and the violation queue weighted by what depends on the affected resource, compares the period's spend against the run rate, and finds resources whose last-seen timestamp says they are gone. It produces one ranked digest per workspace, every finding linked to its record.

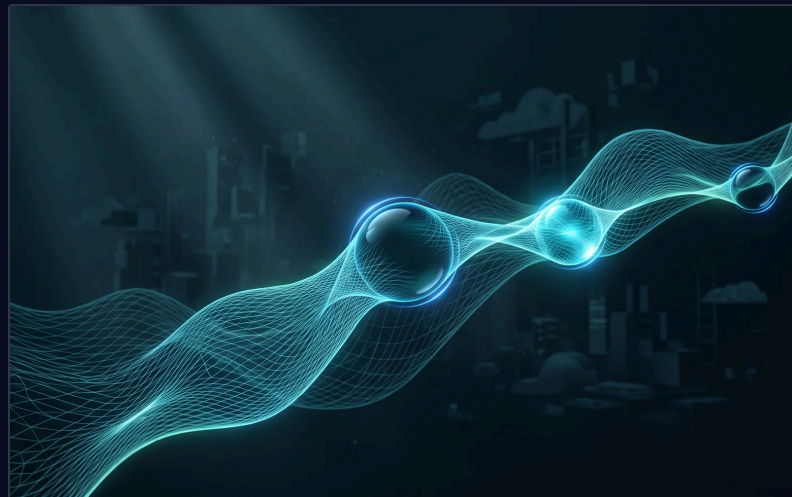
- A deliberate, permanent ceiling: the loop recommends and notifies. It will not remediate drift, waive a violation, disable a connection or resize anything.

06 Built AI-native

1 OF 2

The AI reads the estate you actually have — and it cannot hand back a credential.

Most AI features bolted onto infrastructure tools share two weaknesses. They reason over a description of the estate rather than the estate itself, and they sit next to a credential store that a sufficiently clever prompt might one day talk into leaking. AdapterCloud was built so neither is possible. Inventory, dependencies, policies, violations, drift and cost are all views of one record, discovered by signing in to the real accounts and clusters and writing down what answered — so there is no export, no second index and nothing to fall behind.



The assistant is mounted in the application shell rather than on a page, so it is present across all ten sections without you going to find it. Before your question is sent, the interface resolves where you are into a concrete record: open an adapter, a connection, a resource, a policy or a sync run and “is this one healthy?” needs no identifier pasted. It then works through exactly the same permissioned interface your clicks use — no elevated path, no service key, no second copy of your data. Credentials are structurally out of reach: a connection stores only a reference to a vault secret, credential-shaped values are encrypted when written, and every read path replaces them with a redaction marker.

The reasoning layer — triage, remediation plans, guardrails from plain language, cost anomalies — is designed and committed but not built, and we would rather say so than imply live optimisation. When it lands, it will return the same shape every time: a recommendation, a confidence signal, a rationale, and evidence as links back to real records. Applying anything goes through the audited operations that already exist. Recommending will be a permission of its own, granted separately from acting, because in infrastructure the difference between a suggestion and an unattended change at three in the morning is the difference between a saving and an outage.



Why AI-native, concretely



One live record, not five copies

Inventory, dependencies, policies, violations, drift and cost are all views of the same estate record. Nothing to export, nothing to merge, nothing to fall behind.



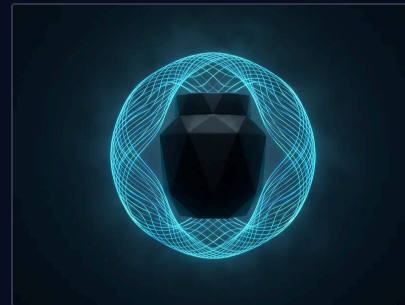
Discovered, not typed in

Collectors sign in to your real accounts and clusters and record what answered, so the AI reasons over the estate you have rather than the one somebody documented.



Governance already has structure

A policy carries a rule and a mode, a violation a severity and evidence, a drift record the exact setting and its expected value. That is structure to reason over, not prose to summarise.



It reads the estate, never a key

A connection holds a reference to a vault secret, never the secret. Credential fields return redacted on every read — by design rather than by instruction.

An assistant on every screen

One assistant, present on the overview, every adapter and connection, the inventory, the topology map, the policy editor, the violation queue, the drift board, run history and cost alike.

Shipped. It opens in a private space of its own that starts when you need it, stays warm as you move between screens, and shuts down when you are done.

Built for your estate, not a generic chat box

It knows it is operating an infrastructure control plane, carries the product's vocabulary, and its usage counts against AdapterCloud rather than a shared pool.

Shipped. You can say “this connection” instead of describing it.

Page context resolved to a named record

Adapter, connection, resource, policy and sync-run pages let you say “this one” and have the request carry that exact record.

Partial, and stated as such: five of ten sections do this. On the violation queue, drift board, topology map and cost view, name the record you mean.

Per-turn metering

Before each turn your credit is checked; afterwards the model that answered and the tokens it used are recorded against your workspace and session.

Shipped. The in-app check is advisory so a slow check never costs you an answer; the hard limit is applied centrally.

37 tools for your own AI client

Adapters, connections, inventory, topology, policies, violations, drift, runs and cost, reachable from any client that speaks the Model Context Protocol.

Shipped as code, connected once with your own sign-in, so the agent sees exactly what your account may see.

Four read-only estate snapshots

The live schema, an adapter snapshot, a connection snapshot and a cost snapshot, offered as resources rather than as tools.

Shipped. An agent can orient itself before it acts, instead of probing with writes.

Two guided agent workflows

Connection onboarding walks from registering an adapter through to a populated inventory. A governance sweep goes the other way: degraded connections, new drift, open violations, spend by service.

Shipped. Ask an agent to run one and it follows the steps in order rather than guessing which comes next.

Credential redaction on every read

Ask about a connection and you get its target, region, sign-in method and health — and a redaction marker where a key would be.

Shipped. The interface cannot emit what it never holds, so this is not a prompt instruction that can be argued with.

Real discovery behind the answers

Cloud accounts and container clusters are polled directly today, with counts and warnings recorded on each run. Classes of system without a collector fall back to the manual path rather than to an error.

Shipped for two collectors. The AI is reading discovered facts, not seed data.

Joined-up estate traversal

Adapters, connections, resources, policies and runs are linked by real relationships, so one question can travel across them.

Shipped. “What did this connection bring in, and when was it last refreshed?” is one question rather than three lookups joined in your head.

Deterministic blast radius

Pick a resource and see what depends on it, directly and downstream, by walking recorded dependencies.

Shipped, and deliberately not badged as AI. It is arithmetic over recorded facts, complete only as far as discovery has reached.

A product playbook for the assistant

The house vocabulary and the usual next question — adapter kinds, connection health, drift paths, enforcement modes — written down for the assistant to use.

Planned. Today the assistant knows where you are but has no product playbook, which is the gap between correct answers and sharp ones.

Violation triage and drift fix plans

Group open violations by shared root cause, rank them by what the resource is and what depends on it, and return an ordered fix per drift record naming the setting and its expected value.

Planned. It will arrive as a ranked queue with a plan attached, and nothing will change until you accept one.

Guardrails written in plain language

Describe a rule in a sentence and get back a draft policy — kind, mode, severity, scope and expression — as editable structure, with anything unresolved listed plainly.

Planned. Before you save, a dry run will show exactly which of your resources the draft would flag.

Cost anomalies, forecasts and rightsizing

Flag a period-over-period jump against the resource that caused it, project the period from the observed run rate, and rank rightsizing candidates by saving against risk.

Planned, with a permanent ceiling: never an automatic resize or termination. The finding arrives with the resource, its dependents and its open violations attached.

Permissions checked outside the AI

The assistant and any external agent carry your own identity, so they see exactly what you see. Anything outside your workspace is not merely hidden — it is not reachable.

An interface that cannot emit a credential

Connections hold a vault reference rather than a secret, credential-shaped fields are encrypted on write, and every read path redacts them. The real value is used only during discovery.

Agent writes audited like manual ones

A change an agent makes is recorded exactly as one you make, with sensitive values stripped from the audit payload — so a change board sees one trail, not two.

A stop switch without a release

All agent write operations sit behind a workspace-scoped flag an administrator can switch off instantly, with no deployment and no downtime.

Metered per turn and attributable

Each answer reports the model that produced it and the tokens it consumed against your workspace and session, with a hard limit applied centrally behind it.

Advisory preflight, central hard stop

The credit check that runs before a turn is deliberately advisory, so a slow or failing check degrades to an answer rather than to a refusal. Exhaustion is enforced centrally, where it cannot be overspent.

Recommending separated from acting

In the designed decision layer, the right to propose is its own permission. A governance lead can hand out advice without handing out authority over the estate.

Proposals carry their evidence

Every planned proposal returns a recommendation, a confidence signal, a rationale and evidence as links back to real records — so you check the reasoning rather than trust the prose.

No unattended infrastructure change, ever

The scheduled sweep and the cost layer recommend and notify. They will not remediate drift, resolve or waive a violation, disable a connection, resize or terminate anything. This is a ceiling, not a first phase.

Volume caps and workspace opt-out

The planned scheduled loop is capped per run so one sweep cannot exhaust a workspace's credits, skips quiet and empty workspaces, and leaves cadence and opt-out with the workspace.

Honest gap: destructive tools are not confirmation-gated

Archiving an adapter or a policy, disabling a connection, waiving a violation, remediating drift and cancelling a run are open to an agent with write scope today, with gateway permissions as the only control. Tool-level gating and an explicit allow-list are governance work still ahead of us.

Honest gap: no run ledger or evaluation set

There is no per-decision history recording what ran, what it proposed, what it cost and what you did about it — and no automated evaluation set for the AI behaviour. A change board will want both, and neither exists yet.

Reachable from, and in numbers

REACHABLE FROM

- The in-app assistant, mounted in the application shell and present across all ten sections
- Adapter, connection, resource, policy and sync-run detail pages, where your question carries that exact record
- Any client that speaks the Model Context Protocol, using your own device-code sign-in
- Guided agent workflows for connection onboarding and for a governance sweep
- Read-only estate snapshots, so an agent can orient itself before acting
- The same permissioned control-plane API your own scripts and pipelines use
- Planned: proposal cards on the violation queue, the drift board, the policy form and the topology inspector — no new screens invented
- Planned: a scheduled per-workspace estate review delivered as a notification digest

IN NUMBERS

- 37 AdapterCloud operations are registered as tools over the Model Context Protocol
- Four read-only estate snapshots and two guided workflows ship alongside them
- Five of ten product sections resolve the page you are on to a named record today
- Ten classes of system are modelled as peers, from public cloud to connected-device fleets
- Five typed dependency edge kinds underpin the blast-radius computation
- Six policy kinds and three enforcement modes are available to author against
- Two provider collectors — cloud accounts and container clusters — perform real discovery today
- Every assistant turn reports the model that answered and the tokens it consumed

What the AI does not do today

What the AI does not do today, stated plainly. There is no inference anywhere in the product's own service — the schema is ordinary tenant-scoped operations with an access requirement on every field, and not one reasoning call. Violation triage, drift remediation plans, plain-language guardrail authoring, cost anomaly detection, forecasting, rightsizing, edge inference and the scheduled estate review are all designed and committed but not built, and they wait on a shared platform reasoning runtime rather than on a model gateway smuggled into an infrastructure service. Page grounding covers five of ten sections. There is no product playbook for the assistant yet, no confirmation gate on destructive agent tools, no per-decision run history and no automated evaluation set. The model is not selectable and bring-your-own-key is not available; a picker is planned. And the ceiling on autonomy is permanent rather than provisional: the AI will not change your cloud accounts, because AdapterCloud discovers and governs — it does not provision.

Six roles are seeded into every workspace, and the interesting part is what each one is deliberately denied. A role that can build the fabric cannot author the policies that judge it; a role that can judge the estate cannot reshape it.

Fabric Owner

The platform lead accountable for the whole connected estate — the only role that can archive an adapter or retire a policy, and the only one that reaches the compliance records at full authority.

- Define and archive adapters
- Author and retire policies
- Reach the compliance queue
- Configure and retire connections
- Resolve or waive violations
- Own the estate end to end

Governance Officer

Authors the rules and works the queues. Holds full authority over policies, violations and drift — and is held at read-only on adapters, connections, resources, topology, runs and cost, so it can see the estate it judges but cannot reshape it.

- Author policies across six kinds
- Record and remediate drift
- Read the whole estate
- Acknowledge, resolve and waive
- Review the compliance trail
- Cannot change what it polices

Fabric Engineer

The platform and integration engineer the product is built for — onboards an account, a cluster, an edge site or a tenant and maps what depends on what. Can create and update adapters but not archive one out of the shared catalogue.

- Define adapters per class of system
- Register the resource inventory
- Start and drive sync runs
- Configure authenticated connections
- Map dependency edges
- Record drift findings

FinOps Analyst

Runs cost-ingest passes and attributes spend to the live inventory, without the ability to change the estate. Writes only to cost records and the runs that produce them.

- Ingest cost records
- Pull cost summaries by service
- Scope spend to one connection
- Start cost-ingest runs
- Read inventory and connections
- Cannot alter the estate

Estate Operator

Keeps the picture current day to day — starts and closes discovery and reconcile passes, triages drift and works violations during incidents, while adapter definitions, connection credentials and policy authoring stay elsewhere.

- Start, complete, fail, cancel runs
- Triage the drift board
- Work the violation queue
- Update the resource inventory
- Read adapters and connections
- Read policies and cost

Estate Auditor

Read-only across the whole infrastructure picture for internal audit, a service-provider client contact or a manager — and deliberately denied the compliance records, which carry personal data that infrastructure review does not require.

- Read the inventory
- Read the topology
- Read policy posture
- Read violations and drift
- Read sync history
- Read spend

Under the hood

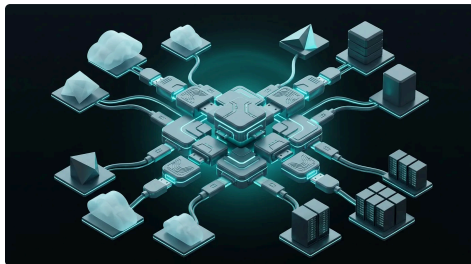
AdapterCloud is a control plane, not an agent you install on every box. It reaches your systems through authenticated connections, records what it finds, and holds the whole model — fabric, governance and operations — in one tenant-scoped store. The product's own screens and your automation call the same interface, so there is no second-class integration surface to keep in step.

The whole control plane is a single graph API. That matters practically: types are consistently namespaced so it composes into a larger enterprise graph without name collisions, and one request can travel from a connection to the resources it brought in to the policies that judge them. Identity and authorisation are validated at the edge, before a request reaches the service — the service itself never sees or validates a token, and never re-implements authorisation.

Discovery is idempotent by construction. Each resource is registered against its provider-native identifier, so a repeated run updates the existing entry rather than creating a duplicate, and the inventory converges. Two provider collectors ship today — public cloud accounts and container clusters — which authenticate, page through the target, upsert what they find along with the dependency edges they understand, and set the connection's health from the real outcome. Other classes of system are fully modelled and driven by a caller reporting results; their collectors are next.

THE SPLIT THAT MATTERS

Services never authorise. Every operation declares the resource, the action and the scope it needs, and that requirement is enforced at the gateway from the composed schema before the request is forwarded. The service trusts the verified identity it is handed and does nothing else with auth. The practical upshot: there is no code path where a feature quietly forgets a permission check, because the check does not live in the feature.



Adapter model

Ten classes of system as peers, each with a vendor, a version, declared capabilities, an optional configuration schema and a publication lifecycle from draft through published and deprecated to archived. Six sign-in methods for connections, including key, delegated authorisation, service account, certificate and assumed role.

Discovery and reconciliation

Sync runs in four kinds — discovery, reconcile, cost ingest and full — with validated state transitions, start and finish timing, error capture, and counters for resources discovered and reconciled and for drift and violations produced. Scheduled recurring runs are planned; runs today are triggered by a caller.

Graph topology

Typed dependency edges with a kind, a weight, a direction flag and attributes. Forward adjacency is built from recorded edges, and both direct dependencies and the full transitive downstream set are computed for any selected node. Multi-hop traversal beyond that, and inferred edges, are planned.

Governance model

Six policy kinds, three enforcement modes, a severity, a rule expression and a scope selector. Violations and drift records each move through their own validated lifecycle, with a required reason on any waiver and a timestamped remediation. Continuous automatic evaluation and blocking enforcement at change time are planned.

Cost ledger

Period-bounded records with amount, currency, service, period and a structured breakdown, attributed to a resource or a connection, plus a workspace-wide summary grouped by service and optionally scoped to one connection. Budgets, anomaly detection and forecasting are planned.

Access and tenancy

Every field declares its access requirement; every read filters by workspace; every write asserts the record belongs to the calling workspace before it changes anything, so a guessed identifier from another tenant is rejected rather than mutated. Six seeded roles, plus organisation and billing presets.

API and extensibility

One graph API for the product and your integrations, scoped API keys minted in the developer portal with an expiry, a fully custom adapter kind for internal systems, and a tool surface for AI clients over an open protocol.

Platform services inherited

Single sign-on, roles, tenancy, billing and quotas, notifications, comments on records, shared file storage, tagging, search, localisation and a health probe come from shared platform services rather than being rebuilt per product. Some product-specific wiring — notify on violation, comment threads on resources — is planned.

Standards and integrations

OPEN STANDARDS IT SPEAKS

- GraphQL — the whole control plane is one typed schema, not a REST surface plus a bolt-on
- Apollo Federation v2 — composes into a larger enterprise graph with namespaced types and no collisions
- OpenID Connect — enterprise single sign-on from day one
- OAuth 2.0 — device-code sign-in for terminal and AI clients, client credentials for integrations
- SAML and SCIM — enforced single sign-on and user provisioning at Enterprise
- Model Context Protocol — the tool, resource and prompt surface for AI clients
- JSON Schema — adapter configuration contracts, policy expressions and structured evidence
- Webhooks — outbound delivery of product events into your own systems
- TLS 1.2 and above in transit; AES-256 at rest; AES-256-GCM for application-layer secrets
- CSV and JSON export for audit and analytics extracts

PLUGS INTO

- Public cloud accounts — a shipping collector authenticates and discovers compute, networking, storage and identity resources with their edges
- Container clusters — a shipping collector discovers nodes, workloads, services, ingresses and persistent volumes
- The remaining public clouds — fully modelled today and driven by a caller reporting results; their collectors are next
- Private cloud and on-premises estates, through their own first-class adapter kinds
- Edge sites and connected-device fleets, modelled as peers rather than as an afterthought
- Network fabric and data systems, each a first-class class of system
- SaaS tenants, so the long tail of admin panels joins the same inventory
- Anything else, through a fully custom adapter you define yourself
- Existing service-management and asset systems — feed them a live, discovered record instead of a manual one
- Existing cost tooling — ingest its lines against the same resources, so its numbers land on your estate map

How it deploys

Multi-tenant cloud

The default. Your workspace is isolated on every read and every write, on shared infrastructure, with no setup on your side.

Dedicated tenancy

Available at Enterprise for organisations that need their own isolated environment for regulatory or contractual reasons.

Self-hosted

Available at Enterprise, including air-gapped deployment, for regulated estates and service providers who cannot send data out.

Data residency

Primary hosting is in India today, with a customer-selected region available at Enterprise. Regional controls are contractual, not best-effort.

Backups and recovery

Automated database backups with a rolling hot tier and a longer cold tier, multi-zone deployment for production-tier services, and recovery objectives set contractually at Enterprise.

09 Security & trust

AdapterCloud holds a map of your infrastructure and references to the credentials that reach it, which makes it a high-value target by definition. The controls below follow from that, and the honest posture on certification is stated immediately after.

Authorisation enforced at the edge

Every operation declares the resource, action and scope it requires, and that requirement is checked at the gateway before the request reaches the service. The service never validates a token and never re-implements authorisation, so a feature cannot quietly skip a check.

Architectural tenant isolation

Every read filters by workspace. Every write asserts that exactly one record in the caller's own workspace matches before changing anything, so a guessed identifier from another tenant is rejected rather than mutated.

Secret references, not secrets

A connection stores a reference into a per-environment secret store, never the credential itself. Credential-shaped values in a connection's settings or a resource's attributes are encrypted when written and replaced with a redaction marker on every read path.

Least privilege, reviewed

Role-based access with quarterly access reviews, just-in-time elevation for production, break-glass accounts with full audit, no shared accounts, and multi-factor authentication required for all administrative access.

Encryption in transit and at rest

TLS 1.2 and above on all public endpoints, with strict transport security and modern cipher suites; AES-256 for primary databases, cache where supported, and object storage; AES-256-GCM for application-layer secrets and signing keys.

Key management and rotation

Keys are held in a managed per-environment key-management subsystem with rotation. Customer-managed keys are available at Enterprise on request.

Validated lifecycles everywhere

Connection health, violation resolution, drift remediation and sync-run state all transition through validated state machines. A violation cannot skip to a nonsensical state, and a waiver cannot be recorded without a reason.

Network posture

Production runs with private networking, default-deny network policies and segregation by environment. Public traffic passes a web application firewall before reaching the gateways; internal service-to-service traffic uses internal gateways only. Operator access is brokered through zero-trust sessions with no direct public administrative access.

Secure development lifecycle

Code review on every change, static analysis and dependency scanning in the pipeline, secret scanning, and container image scanning for known vulnerabilities, misconfiguration and a software bill of materials. Query depth and cost limits are applied at the gateway.

Vulnerability management

Continuous scanning of base images and dependencies, with a patch cadence of seven days for critical, thirty for high and ninety for medium, subject to operational risk assessment. Coordinated disclosure is offered with published reward bands.

Logging, monitoring and audit

Centralised logging, metrics and traces. Governance-relevant actions — policy changes, connection edits, file access — are auditable, with application audit logs retained per plan and production action logs retained for at least 180 days for investigation.

Incident response

Documented playbooks with an on-call rotation, severity classification and internal escalation, and customer notification within 72 hours of a confirmed personal-data breach or sooner where law requires. Every incident closes with a review tracked to remediation.

Data minimisation and segregation

Customer data is logically segregated by tenant and enforced in the resolver layer. Service-generated data is aggregated or de-identified before any analytical use. Copying production personal data into non-production environments is prohibited.

Data subject rights and portability

Account exports for personal data and infrastructure records, administrative tooling to correct and delete records, and in-product acceptance and data-subject request records held as first-class entries — reachable only by the two roles that need them, never by a general viewer.

Stated plainly: certification

AdapterCloud holds no security certification today, and we will not imply otherwise. The information-security programme is built against ISO/IEC 27001 control families and the SOC 2 trust services criteria, and the controls above are implemented — but SOC 2 is targeted for an initial report in Q4 2026 and a Type II report in Q2 2027, and ISO/IEC 27001 certification is targeted for Q4 2026, with ISO/IEC 27701 following in Q1 2027. Annual third-party penetration testing is a stated target rather than an established track record. India's data-protection regime is treated as continuous compliance with an external readiness assessment targeted for Q4 2026, and a formal GDPR gap assessment for Q1 2027. Payment card data never touches our systems: card capture is handled by the payment provider's hosted fields. Compliance evidence exports and streaming into a security information and event management system are on the roadmap, not shipped. A data processing agreement is available on request from Growth upward. If a current certificate is a hard procurement requirement for you today, we would rather tell you now than at the end of a security review.

One product, many front doors

The control plane and its API are live. The rest of the ecosystem is sequenced behind launch, and each entry below says exactly where it stands rather than implying it is ready.

Product control plane

Available. The workspace-native web application with ten sections — overview, adapters, connections, resources, topology, policies, violations, drift, sync runs and FinOps — all calling live operations. Remaining frontend work is depth, principally richer cost charting, rather than existence.

Control-plane API

Available. The complete fabric, governance and operations model over one federated graph API, with access requirements enforced at the gateway. It is the same interface the product's own screens use.

In-app AI assistant

Available. Mounted in the application shell so it is present across all ten sections, grounded in the record you have open on five of them, and metered per turn.

AI agent tool surface

Partly available. 37 operations, four read-only snapshots and two guided workflows over the Model Context Protocol, signed in with your own identity. It runs from source today — there is no published package to install yet.

One product, many front doors

Developer portal

Available. Register an application, mint an API key scoped to exactly the permissions it needs with an expiry, and wire it into your scripts or pipelines. Product-specific developer documentation is in progress.

Marketing website

Available at adaptercloud.com, including the product story, the use-case library, comparison pages, published pricing and the waitlist.

Documentation site

In progress. Product guides, API reference and integration tutorials are being written ahead of launch.

Command-line interface

Planned. Terminal-native sign-in and automation for inventory, governance, sync runs and cost workflows across environments.

One product, many front doors

Node and Python SDKs

Planned. Typed access for Node and Bun projects, and an async client for Python automation, inventory pipelines and operational notebooks.

Desktop application

Planned. A native build of the control plane for operators who want it outside a browser tab.

Mobile applications

Planned. Native iOS and Android for reviewing estate health, triaging violations and drift, and approving actions away from a desk.

Editor and browser extensions

Planned. Estate context and governance signals surfaced alongside the code you are changing, and beside the provider consoles you still have to visit.



Five tiers, published openly, with the same data model, tenant isolation and core API on every one of them. What scales is capacity — connections, discovered resources, run concurrency, governance depth, retention, support and service level. The prices below are the primary regional band; other regions are priced lower.

FREE

\$0 forever

Evaluators, micro-businesses and a single platform engineer proving the idea.

- Full product surface at activation-sized quotas
- 250 discovered resources, 2 connections
- 2 admin seats, 1 workspace, 1 concurrent discovery run
- 5 governance policies, 1,000 policy evaluations a month
- 1 GB shared storage, 30-day analytics retention
- No card required, no expiry

STARTER

\$99/mo or \$79/mo annual

A single team or a small estate that has outgrown the free ceiling.

- 2,500 discovered resources, 10 connections
- 5 admin seats, 3 concurrent discovery runs
- 50 policies, 25,000 evaluations a month
- 25 GB shared storage, 90-day analytics retention
- 10 integrations, 30-day audit retention
- Email support with a 48-hour target response

GROWTH

\$399/mo or \$319/mo annual

Scaling multi-cloud and hybrid teams running an actively governed estate.

- 25,000 discovered resources, 50 connections
- 15 admin seats, 3 workspaces, 15 concurrent runs
- 250 policies, 250,000 evaluations a month
- 100 GB storage, 365-day analytics, 180-day audit
- Advanced analytics, custom dashboards, scheduled reports
- Priority support under 24 hours, 99.5% service level

BUSINESS

\$999/mo or \$799/mo annual

Established platform and FinOps functions wanting one predictable monthly price.

- 250,000 discovered resources, 250 connections
- Unlimited concurrent discovery runs, fair use
- 1 TB pooled storage, 730-day analytics, 365-day audit
- 50 admin seats, 10 workspaces
- 1,000 pooled policies, 2.5M evaluations a month
- Priority support under 8 hours, 99.9% service level

ENTERPRISE

From \$12/resource/mo list; \$9 annual at 250+

Large, regulated or multi-entity organisations and managed-service providers.

- Custom capacity across every quota
- Dedicated tenancy, self-hosting or air-gapped
- Audit streaming, custom retention, legal hold, access reviews
- Enforced single sign-on and user provisioning
- Customer-selected residency and customer-managed keys
- Dedicated success manager, 99.95%+ service level

Add-ons rather than forced upgrades

Capacity packs extend a tier without moving to the next one: extra discovered resources, concurrent discovery runs, policy evaluation volume, governance seats, storage, API volume, integrations, delivery volume, advanced analytics, audit retention and prepaid support hours. Add-ons are available from Starter upward; a free account upgrades first.

No silent overage

You are warned at 80% of a quota and blocked at 100%, then offered an upgrade or an add-on. Nothing is billed above plan without your choosing it. After a downgrade, data above quota becomes read-only rather than deleted, until you upgrade again or reduce below the limit.

Regional pricing

Prices are banded by market. The figures above are the primary band; other bands are materially lower — the entry paid tier ranges from \$99 down to roughly a third of that in the lowest band, with local-currency equivalents at checkout. Enterprise floors are banded the same way.

Enterprise shapes

Enterprise moves from a flat fee to a per-resource annual contract, because that is how large estate deals are scoped and renewed. Multi-year and volume terms are available, invoiced net thirty by default. Implementation, migration and architecture consulting are sold separately as statements of work.

One resource record, not modules that need reconciling

Cost lines, drift findings, policy violations and dependency edges all reference the same resource entry. There is no extract-transform-load step between an inventory module and a cost module, because there are no modules — a tag change is visible to policy and to cost immediately.

Ten classes of system as peers

Public cloud, private cloud, on-premises, edge, connected devices, network, container clusters, data systems, SaaS and custom are first-class kinds in one model. Cloud-only and cluster-only tools cannot express half of that estate, and asset-management suites treat the cloud half as an import.

Topology is a data structure, not a drawing

Dependencies are typed edges — depends-on, connects-to, contains, routes-to, replicates-to — with direction and weight, so blast radius is a traversal you can defend rather than a diagram that was accurate in the first quarter.

Governance findings are work items

Drift moves through detected, acknowledged, remediated or ignored; violations through open, acknowledged, resolved or waived, with a required reason on any waiver. Each is a queryable item with a history, not a row in a report that gets regenerated.

Discovery-driven, not process-maintained

Registration is idempotent by provider-native identifier, so re-running discovery converges rather than duplicating, and each pass is a tracked run with counters. Time to a useful inventory is days, not a multi-quarter configuration-database population project.

Cost on the graph you operate

Spend attaches to the resource or connection it belongs to and rolls up by service, on the same estate view as inventory and dependencies. Finance and platform stop maintaining two pictures and reconciling them monthly.

The API is the product, not a side door

The product's own screens call the same graph API your integrations call, with scoped keys and the same audit trail. Namespaced types mean it composes into a larger enterprise graph rather than colliding with it.

Isolation is architectural

Every read is workspace-scoped and every write proves ownership before mutating, so cross-tenant access is not a permission you could misconfigure. That is what makes one operating model across many client estates safe for a service provider.

Honest status on every capability

The website, the changelog and this document mark each capability shipped, partial or planned, including where an alternative is genuinely ahead. You will not discover a roadmap item during your evaluation.

You are pre-launch. Why would we bet on this?

You should weigh that seriously. What is real today is substantial: the control plane and its API are live, the full fabric, governance and operations model persists and enforces tenant isolation, and two provider collectors do genuine discovery. What is not real is equally clear — no certifications, no references to call, and a reasoning layer that is designed rather than built. Start on the free tier, connect one account, and judge the product rather than the pitch.

We already have a cost tool and a configuration database. Is this a rip and replace?

No, and we would not recommend it as one. Deep licence optimisation, cost-optimisation recommendations and service-management workflow are genuinely better in the incumbents, and we say so on our own comparison pages. The sensible pattern is to use AdapterCloud as the live source of record and feed it into what you already run, so those tools operate on current data instead of a monthly export.

Does it provision infrastructure?

No. AdapterCloud discovers, models, governs, observes and attributes cost. It does not provision, and self-service blueprints are explicitly a roadmap item rather than a claim. If provisioning is your primary requirement, a cloud management platform or an infrastructure-as-code platform is the right purchase, and AdapterCloud complements it as the record of what exists — including everything created outside that pipeline.

Only two collectors ship. Is that not a very small footprint?

It is a real limitation and the one we are working on next. Public cloud accounts and container clusters discover automatically today. Every other class of system is fully modelled — it just relies on a caller reporting results through the API, an agent or a script, rather than on a packaged collector. Collectors for the remaining kinds are the next sequenced item.

Automatic policy evaluation is not built. So what does governance actually do?

Today it gives you the structure and the lifecycle: policies across six kinds with modes, severities and scopes; violations with evidence and a validated open-to-waived path; drift records naming the exact setting and its expected value. What is missing is the engine that walks the inventory on a schedule and emits violations without being asked, and blocking enforcement at change time. Both are committed. Until then, violations are recorded through the API or by hand.

Can the AI change our cloud accounts?

It cannot, structurally. AdapterCloud does not provision, so there is no path from the assistant to your infrastructure. It can change AdapterCloud's own records within your permissions if you ask, and each such change is audited exactly like a manual one and sits behind a switch an administrator can turn off without a release. The honest caveat: agent write tools are not confirmation-gated yet, so grant write scope deliberately.

You hold credentials for every system we run. Why is that safe?

We do not hold them. A connection carries a reference into a per-environment secret store, and credential-shaped values in configuration or attributes are encrypted when written and redacted on every read. The real value is used only during discovery. That said, a control plane with reach into your estate is a high-value target by definition, which is exactly why self-hosting and dedicated tenancy exist for organisations whose risk appetite does not extend to shared infrastructure.

We need SOC 2 before we can buy anything.

Then we are not purchasable yet, and we would rather say so at the start of a conversation than at the end of a security questionnaire. The programme is built against the SOC 2 criteria and ISO/IEC 27001 control families, with an initial report targeted for late 2026 and a Type II report in 2027. If your timeline can accommodate a design-partner engagement ahead of that, the controls and the documentation are available for review now.

How do we know this will not become another stale inventory?

That failure mode comes from inventories maintained by process. This one is maintained by discovery, registered against provider-native identifiers so repeated passes converge, with every pass recorded as a run with timing, errors and counters — so staleness is visible rather than assumed. The honest gap is scheduling: runs today are triggered by a caller or by you, and recurring scheduled discovery is a committed roadmap item.

Our estate is mostly one cloud. Is this overkill?

Possibly, and if a single provider's own tooling answers your four questions, use it. AdapterCloud earns its place when the estate crosses a boundary — a second provider, a data centre, a cluster fleet, edge sites, a set of SaaS tenants — or when governance and cost need to sit on the same record as operations. The free tier exists so you can find that out at your own pace.

Where it sits against the alternatives

VS CLOUD MANAGEMENT AND PROVISIONING PLATFORMS

They lead on self-service provisioning catalogues and blueprints, and on deep hypervisor integration — genuinely, and we do not compete there at all. AdapterCloud leads on the unified record: one resource entry shared by inventory, topology, governance and cost, with no module-to-module reconciliation, and edge, connected devices and SaaS tenants as first-class kinds rather than extensions.

VS CLOUD COST MANAGEMENT AND FINOPS SUITES

They are ahead on optimisation recommendations, budgeting and forecasting, and will remain so for a while — ours are on the roadmap, not in the product. What they cannot do is tell you what a line item depends on, because cost sits detached from an operational topology. AdapterCloud attaches spend to the same resource you operate and govern, and its price does not rise with your cloud bill.

VS IT ASSET MANAGEMENT AND CONFIGURATION DATABASES

They win on asset lifecycle, service-management workflow, audit readiness at enterprise scale and reference-data enrichment. Their weakness is that the record is kept current by process and integration rather than by discovery, which is why population projects run for quarters. AdapterCloud reaches a live, graph-native record in days and can feed theirs — but it is not an asset-lifecycle or licence-optimisation product, and it is not planned to become one.

VS INFRASTRUCTURE-AS-CODE PLATFORMS

They own provisioning, state and plan-time governance, with an enormous provider ecosystem, and we are not building any of that — it is explicitly not planned. Their model is desired state; ours is what is actually running, including everything created by hand, by a legacy tool or through a SaaS admin panel. Keep the pipeline for apply, and use AdapterCloud as the record over the whole estate.

VS CONTAINER-CLUSTER COST TOOLS

They are deeper on cluster-level allocation and optimisation than we will be, and the sensible pattern is to ingest their output into AdapterCloud so cluster spend joins the rest of the estate. Their scope is one class of system; ours is the unified graph in which that spend sits next to the cloud, on-premises, edge and SaaS resources it actually depends on.

VS BUILDING IT YOURSELF ON INTERNAL SCRIPTS

An internal script is faster to start and shaped exactly to your estate, and for a small single-provider footprint it is often the right answer. The cost arrives later: rate limits, a dependency graph drawn by hand, governance living in pipeline checks, and a growing pile of glue that one person understands. AdapterCloud is the API-first version of what that script becomes, with tenancy, roles and audit already solved.

START TODAY

Connect one account and see what you actually run

AdapterCloud launches in 2026 and is in waitlist mode today. The free tier needs no card, and the honest status of every capability is published rather than discovered during a trial.

- Join the waitlist and start on the free tier — full product surface, 250 discovered resources, no card
- Define an adapter for one class of system and connect a single real target
- Run discovery, record a few dependency edges, and author one tagging policy
- Book a walkthrough if you want the estate model mapped to your own environment before you commit

JOIN THE WAITLIST

adaptercloud.com

BOOK A DEMO

adaptercloud.com/contact

SEE THE AI STATUS MATRIX

adaptercloud.com/ai

TALK TO SALES

sales@adaptercloud.com

RESPONSIBLE DISCLOSURE

security@adaptercloud.com
