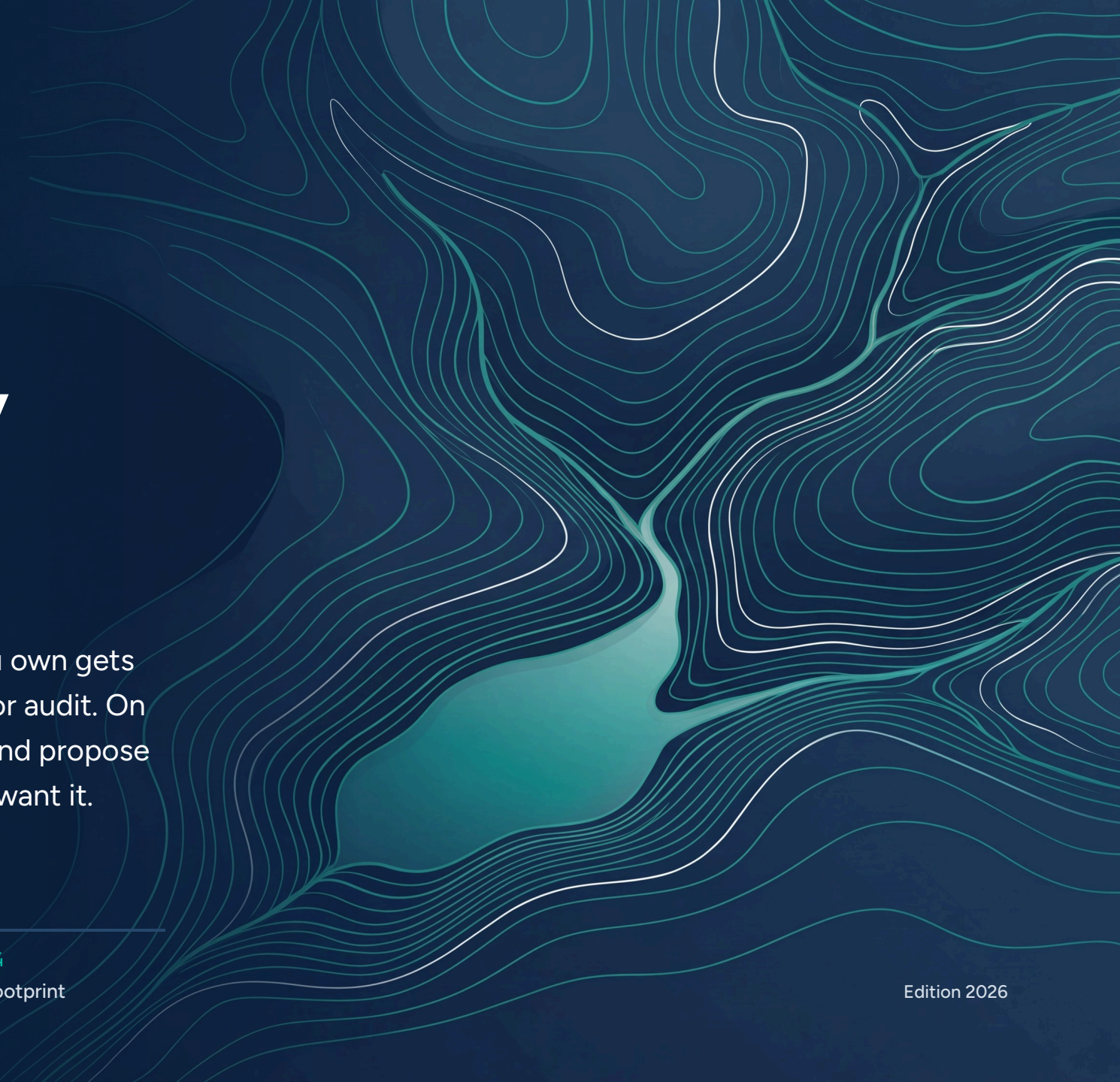
Logmetry

The Logmetry Blueprint

Your SIEM and APM get what they need, a Lake you own gets the clean copy, and an archive keeps the raw one for audit. On that clean data, agents you own triage your alerts and propose the fixes. A smaller footprint comes last, when you want it.



01 Foundation

02 AI triage

03 Remediation

04 Footprint

Edition 2026

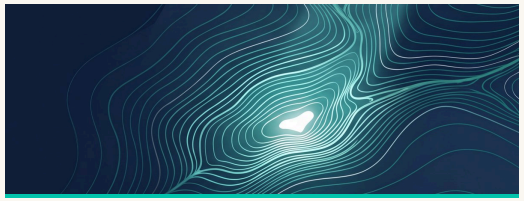
Control and clean the data first, put AI to work on it, fix at the speed of AI, then shrink the footprint.



01

Foundation

Take control: clean, enrich and route every source. A clean Lake and a raw archive you own.



02

AI triage

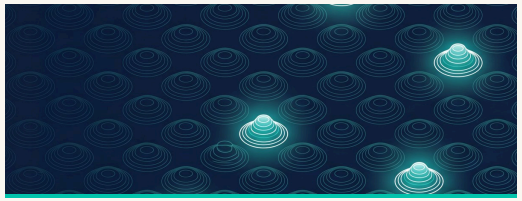
SOC agents you own work your alerts, on the clean data, read-only, with a person on every exception.



03

Remediation

Agents fix what they find, as pull requests a person approves. Everything is code.



04

Footprint

Collect once on an open standard, and shrink the expensive tools to what earns its place.

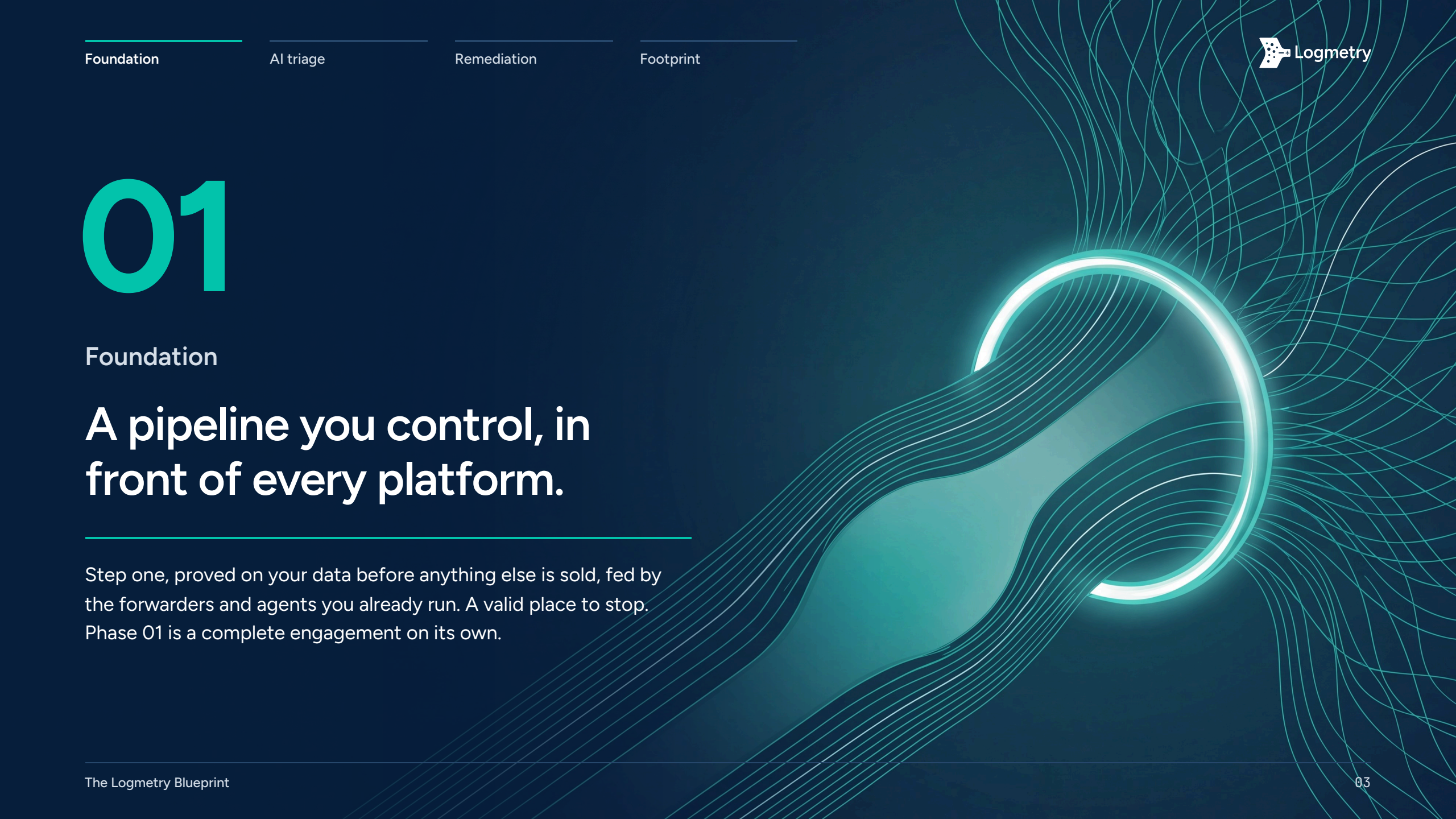
Each phase is sold only after the one before it is proved. Everything as code and AI native run through all four.

01

Foundation

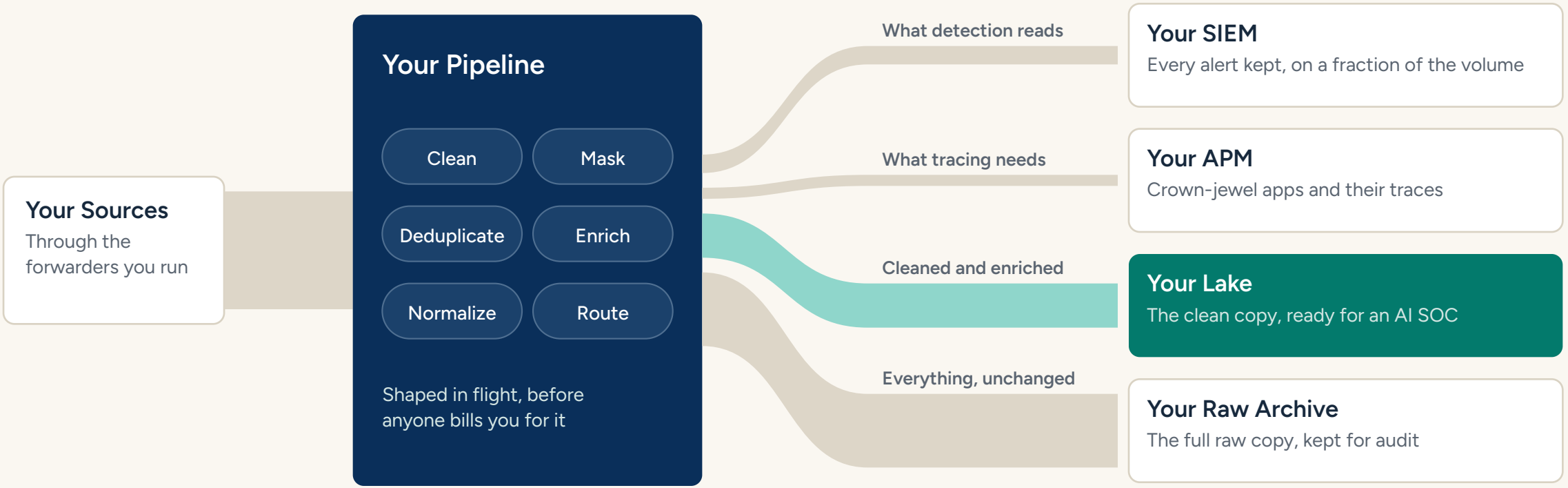
A pipeline you control, in front of every platform.

Step one, proved on your data before anything else is sold, fed by the forwarders and agents you already run. A valid place to stop. Phase 01 is a complete engagement on its own.



Routed four ways: your SIEM and APM get what they need, the Lake the clean copy, the archive the full raw copy.

Exhibit 1 Band width is volume



Every alert kept, more coverage. PII is masked before any model sees it, and the spend cut from noise pays for sources the SIEM never saw.

One Windows server, five minutes: twelve thousand lines arrive, and six are worth an alert.

Exhibit 2 One Windows server's security log over five minutes. Example counts

What arrives	Lines	Your SIEM	Your Lake	Raw archive
Servers logging on to each other Event 4624, automatic, all day long	1,240	Dropped. No alert uses it	One summary row	Kept
The same servers logging off Event 4634	1,236	Dropped. No alert uses it	One summary row	Kept
Normal connections the server allowed Event 5156, the busiest line of all	8,900	Dropped. No alert uses it	Counted, per app	Kept
Exact copies of the lines above Sent twice by two collectors	1,200	Removed	Removed	Kept, as received
Someone typing the wrong password Event 4625, six times for jsmith	6	One alert, with who and where	Six clean rows	Kept
In five minutes From one server	**12,582**	**1 event**	**20 rows**	**12,582**

Nothing is lost. The archive keeps every line as it arrived, for the auditor. Your SIEM gets the one thing an alert needs, already explained, and your Lake a clean summary for AI and replay.

A firewall line in plain words, and the pattern an alert actually fires on.

Exhibit 3 One Cisco firewall line, then five minutes of that firewall. Example counts

```
%ASA-4-106023: Deny tcp src outside:203.0.113.45/52144 dst inside:10.0.2.15/3389 by access-group "outside_in"
```

- A Cisco firewall block
- What happened
- Who: an address on the internet
- Target: a server, remote desktop
- The rule that blocked it

Arrives in five minutes

104,340 lines

104,000 ordinary connections opening and closing

340 blocks like the line above, from one address

Your SIEM gets

1 event

The 340 blocks folded into one

+ the target is a domain controller

+ the address is on a threat list

An alert fires: someone on the internet is trying remote desktop on a key server

Your Lake gets

400 rows

Ordinary traffic rolled up: one row per server, every five minutes

The 340 blocks, cleaned

Raw archive keeps

104,340 lines

Every line as it arrived, for the auditor

One blocked connection means little. Three hundred and forty from one address against a domain controller is an attack, and that is the one event your SIEM receives.

One alert, enriched in flight: the four lookups an analyst makes by hand arrive already done.

Exhibit 4 The six wrong passwords from exhibit 2 and the blocked address from exhibit 3, as one alert

Today, as your SIEM fires it

```
rule: multiple failed logons
user: jsmith
host: DC01
count: 6 in 2 min
src.ip: 203.0.113.45
```

Then the analyst looks up, by hand:

- The user's groups, in the directory
- The host's role, in the asset list
- The address, in threat intelligence
- The address, in the firewall and Linux logs

Enriched in flight, by the pipeline

```
rule: multiple failed logons
user: jsmith
+ user.privileged: true
+ host.role: domain controller
+ src.ip: threat list, outside
  your regions
+ related: 212 sshd failures,
  340 firewall denies, same IP
+ next: 4624 type 10, RDP,
  jsmith on DC01 at 14:06
```

Phase 02, what the agent appends

Verdict.
Likely credential attack, and one success: an RDP logon on DC01 at 14:06.

Evidence.
Six failures, then the logon, from the address behind 212 sshd failures and 340 firewall denies.

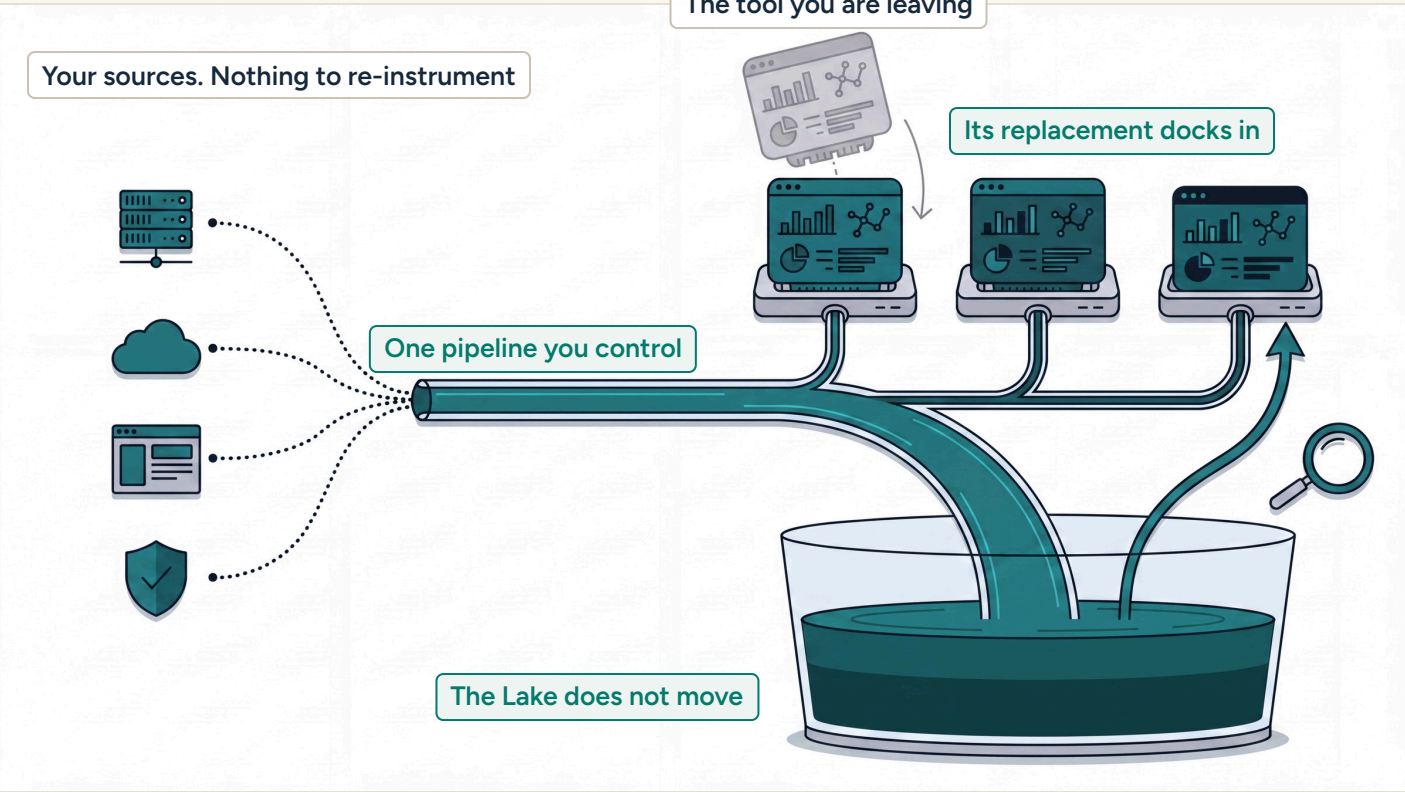
Ruled out.
Travel and a known scanner.

Proposed.
Block the address at the firewall, as a pull request a person approves.

The same shape holds for an APM alert: a latency spike arrives with the deploy, the owning team and the last change already on it.

Adding, replacing or leaving any tool becomes a routing change rather than a migration.

Exhibit 5 A destination swapped, the Lake unmoved



The tools you pay the most for should be the easiest ones to leave. Send each platform only what it is good at, keep the clean copy in a Lake you can replay from, and negotiate every renewal after that from strength.

Search the Lake, replay anywhere. Clean, enriched and partitioned, it stays the ground for triage and search, and replays history into whatever you plug in next.

A valid place to stop. Phase 01 is a complete engagement on its own. None of it needs us in the room to keep running.

Logmetry

02

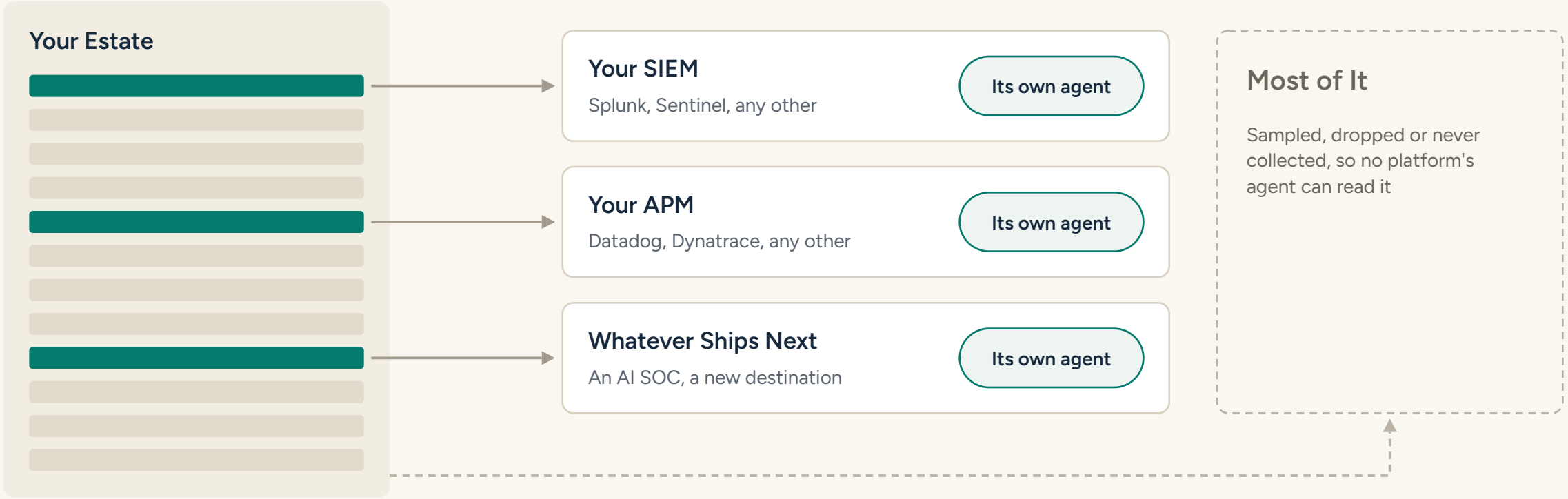
AI triage

SOC agents you own, on the clean data.

Root cause, threat hunting, and the alerts your team never gets to. Comes straight after Phase 01, once it is proved on your data. A person on your team reviews every exception.

Every platform now ships its own agent, and each one reasons over only that platform's copy of your estate.

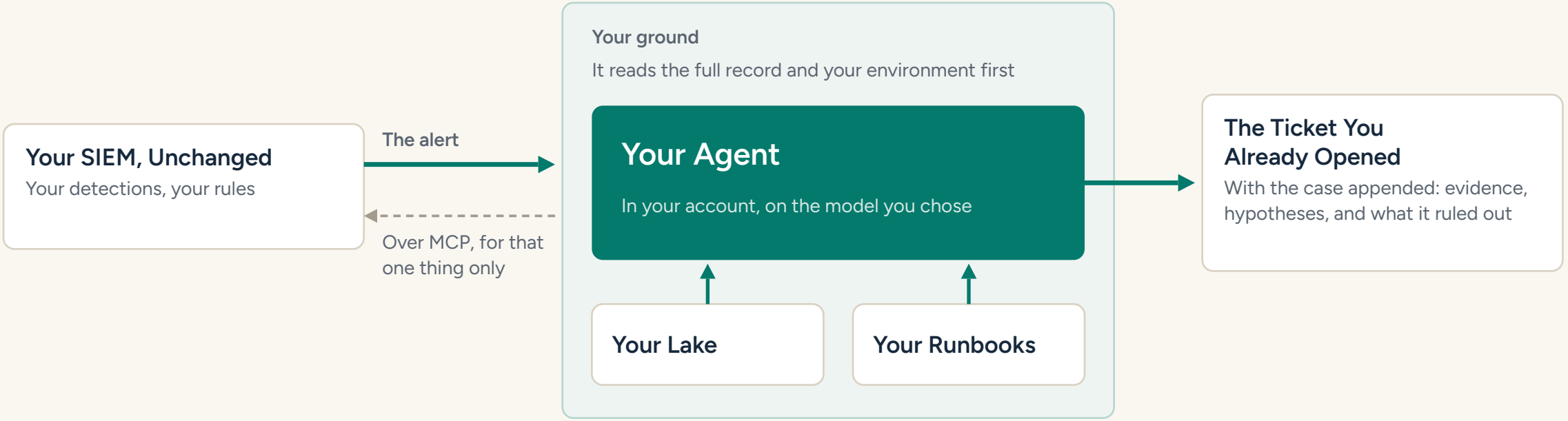
Exhibit 6 A slice each



Each one reasons over what earned its way into that licence, not the full record and not the sources that went somewhere else. Keep a platform's own agent for the simple alerts it can close on its own.

Your SIEM stays. What moves is where the thinking happens.

Exhibit 7 The alert, worked on your ground



How the Investigation Runs

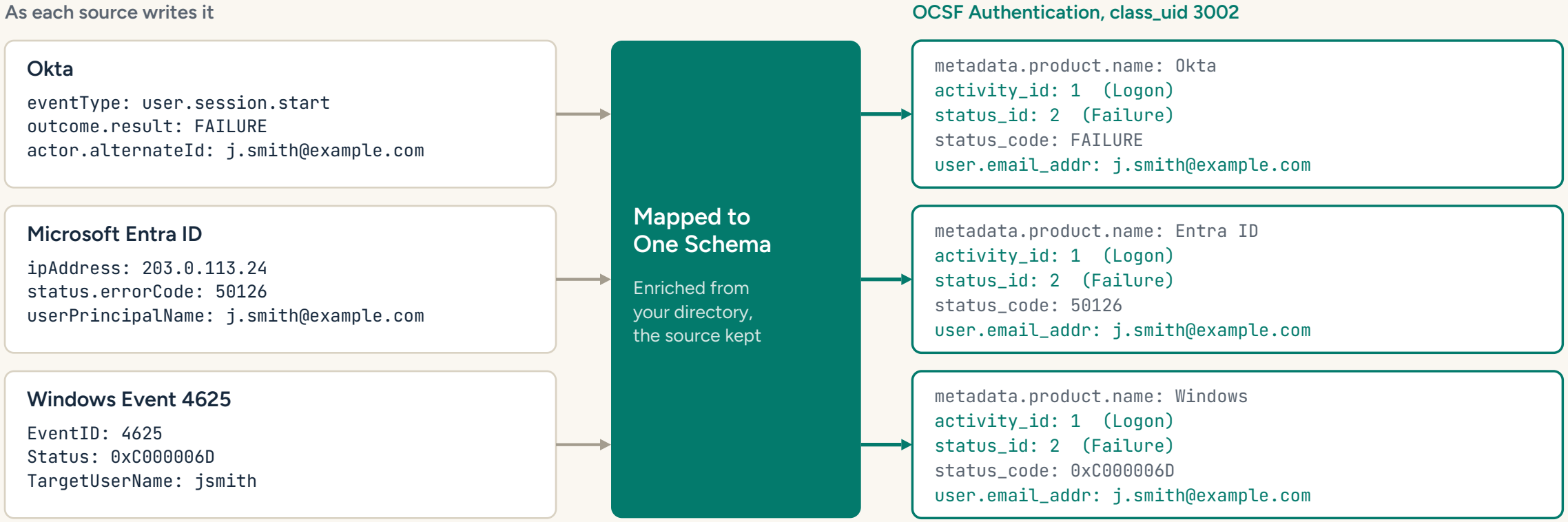
The alert reaches your agent directly. It reads your Lake first, because that is where the clean, enriched history lives, and your runbooks, because that is where your environment is described. When it needs something only the platform holds, it calls that platform for that one thing over the platform's own MCP server, and where a platform does not publish one we build it. One route per platform, used for what only that platform has.

What Lands on the Ticket

Everything the investigation found is appended to the ticket your service desk already opened. Not a verdict on its own. The evidence, the hypotheses it tested, the ones it ruled out and the reason it ruled them out. Your analyst opens one ticket and reads a case rather than starting one.

Every source describes a failed login the same way, about one person.

Exhibit 8 The same failed login from three sources, each mapped across to the same fields

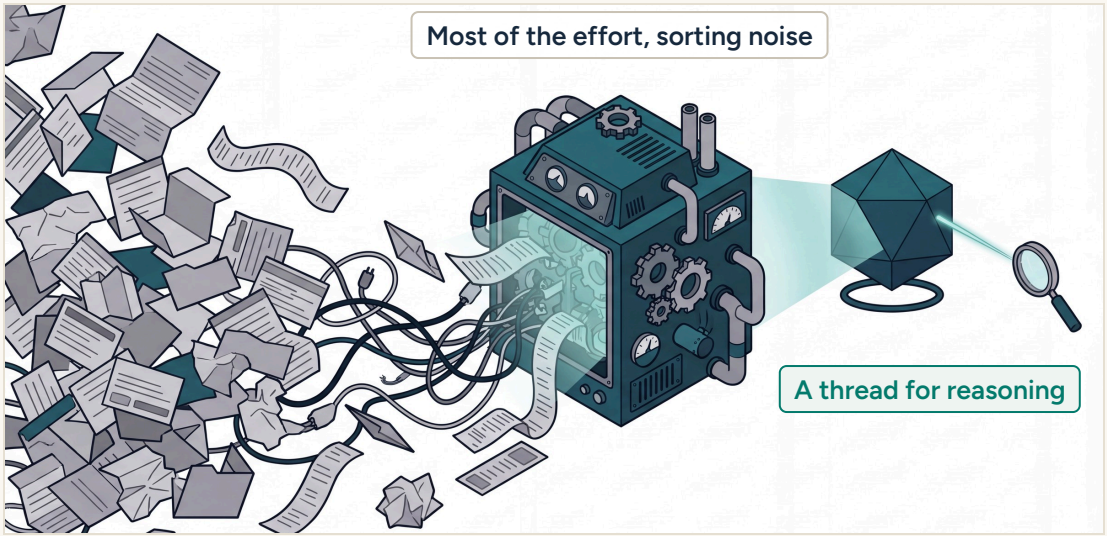


Security events are mapped to OCSF, the open security schema, and joined on the same host, user and service. The agent reads one shape for every source, so the same model reaches better verdicts and every investigation costs less.

Noise, duplicates and boilerplate removed, the agent spends its tokens on what triage needs.

Exhibit 9 One Windows logon event and the same agent. Bars to scale: teal kept, grey removed, bright teal added context

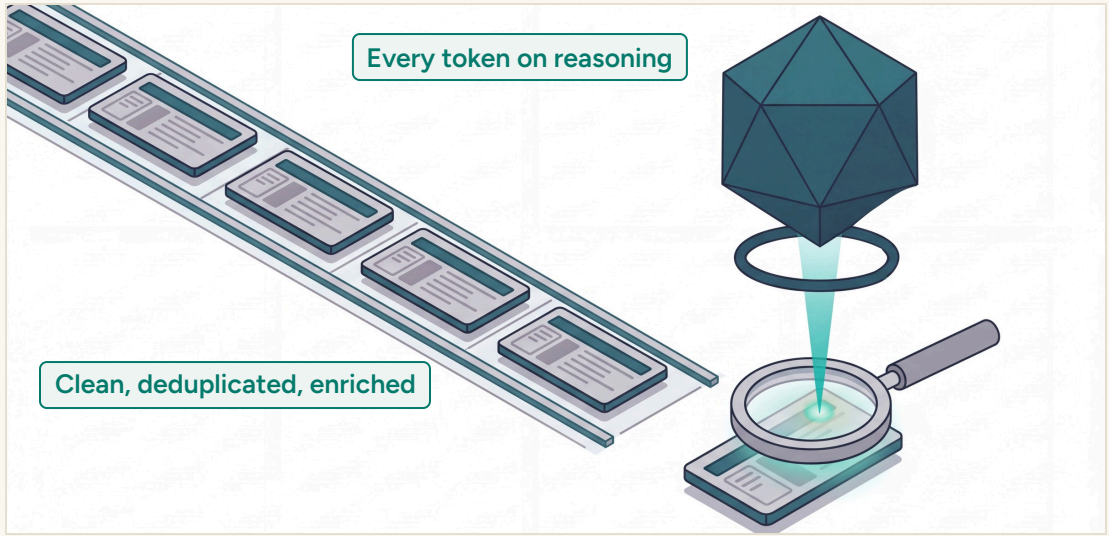
Before



579 removed 710 tokens

Before. One Windows logon event as raw XML: schema boilerplate, empty fields, an all-zero ID and codes nobody triages on. In a real hour, thousands of near-identical lines and duplicates come on top.

After

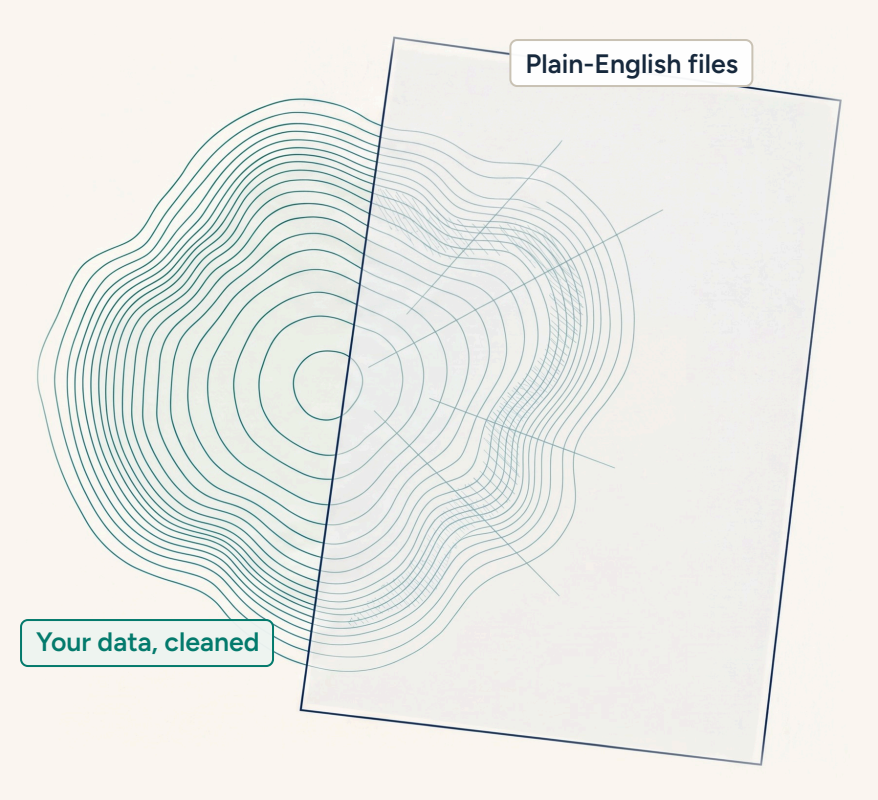


152 tokens

After. Noise dropped, duplicates removed, routine activity rolled up, and the lookups an analyst would make added: host role, account value, working hours. Microsoft's sample event 4624, counted with tiktoken.

Context is the half of every investigation nobody can sell you, because it describes your organisation.

Exhibit 10 Your data, and the files that explain it



Your data, cleaned, deduplicated, enriched and in one schema, so your agents spend their context reasoning rather than translating field names.

Plain-English files that explain how the pieces relate: which service depends on which, who owns what, and what normal looks like on a Tuesday.

Tribal knowledge, written down. The dependency nobody drew, the escalation only one engineer remembers, the reason a service is set up the way it is. We write it with you as the agent's own instructions, so it reasons like a manager who knows how everything connects. No platform can ship that. It takes someone working inside your business, one to one, to build it.

The data without the explanation is a haystack, and the explanation without the data is a wiki.

A small model works the endless volume, and the best reasoning is saved for the investigations that earn it.

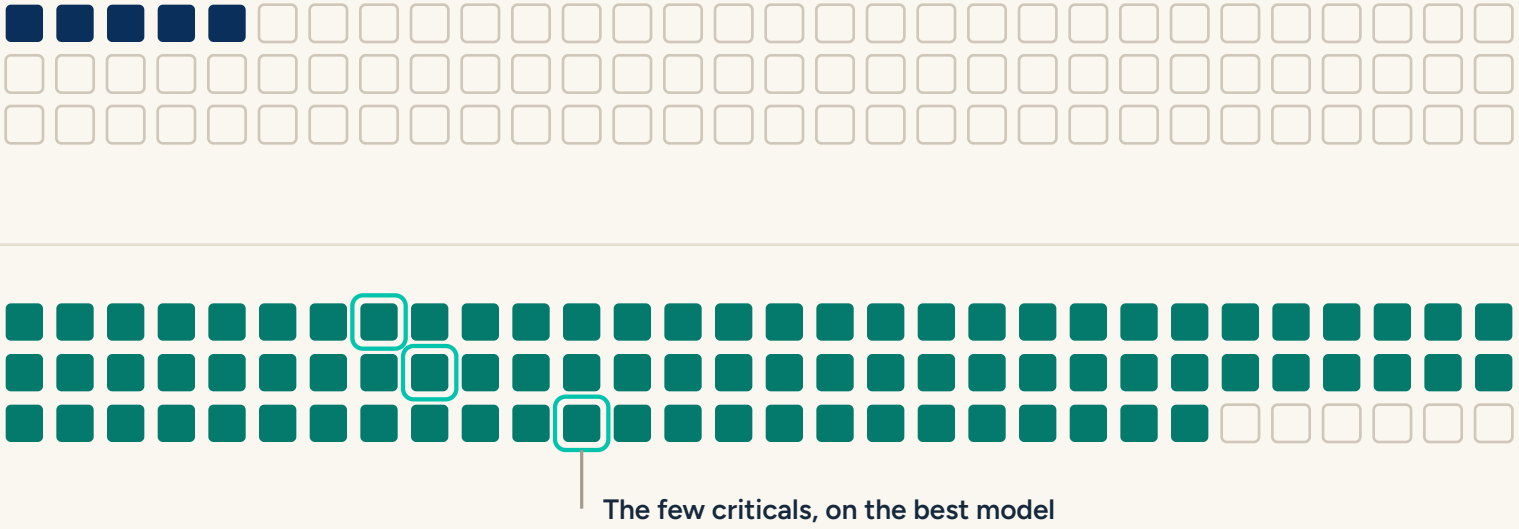
Exhibit 11 Every square is an alert. Illustrative, not a measured ratio

What gets looked at today

The top of the queue, and the rest closed unread

What gets looked at instead

The alerts you hand it, by a small model against your Lake, and the best model for the few criticals



The volume. Low and warning alerts, lookups and retrieval, the ones closed unread today, worked all day by a small model against the Lake.

The criticals. Correlations that cross four sources and a change record, where a wrong answer costs somebody a night.

The model is yours. Swap it the day something better ships, with an edit in a repository rather than a renegotiation.

03

Remediation

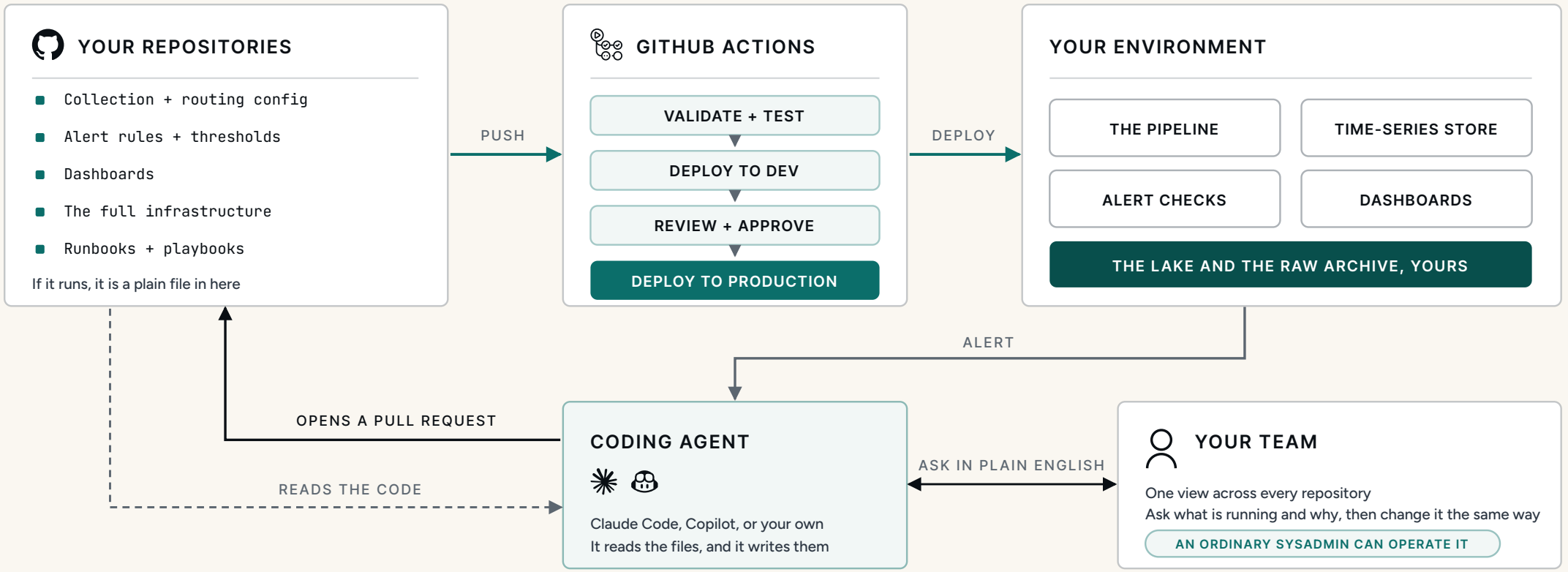
Agents fix what they find, as pull requests.

Everything is code, so the agent that found the cause writes the correction. Comes after AI triage is working your alerts. The agent never changes production on its own: a person approves every pull request.



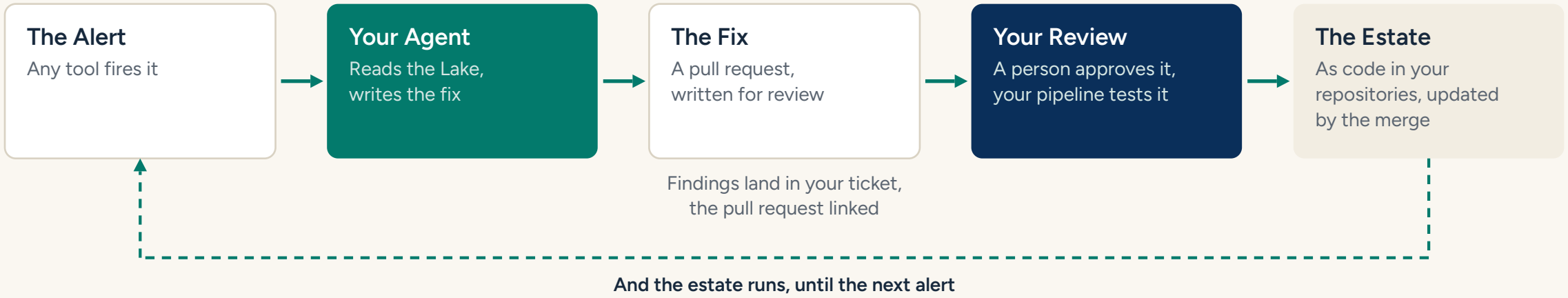
Everything we build is code you own, so an agent can read it and propose the fix.

Exhibit 12 Every change takes the same road



From the alert to a merged fix at the speed of AI, and nothing reaches production until a person on your team merges it.

Exhibit 13 Find it, fix it, review it, ship it



The agent that worked the alert already holds the evidence, so the fix arrives with its reason attached. Your team reviews a change, not a mystery, and the history of every fix is the history of your merges.

What the agent sends. One pull request: the fix as code, with the evidence and the causes it ruled out.

What your team keeps. The merge. Your branch rules and your CI decide what reaches production, as they do for every change today.

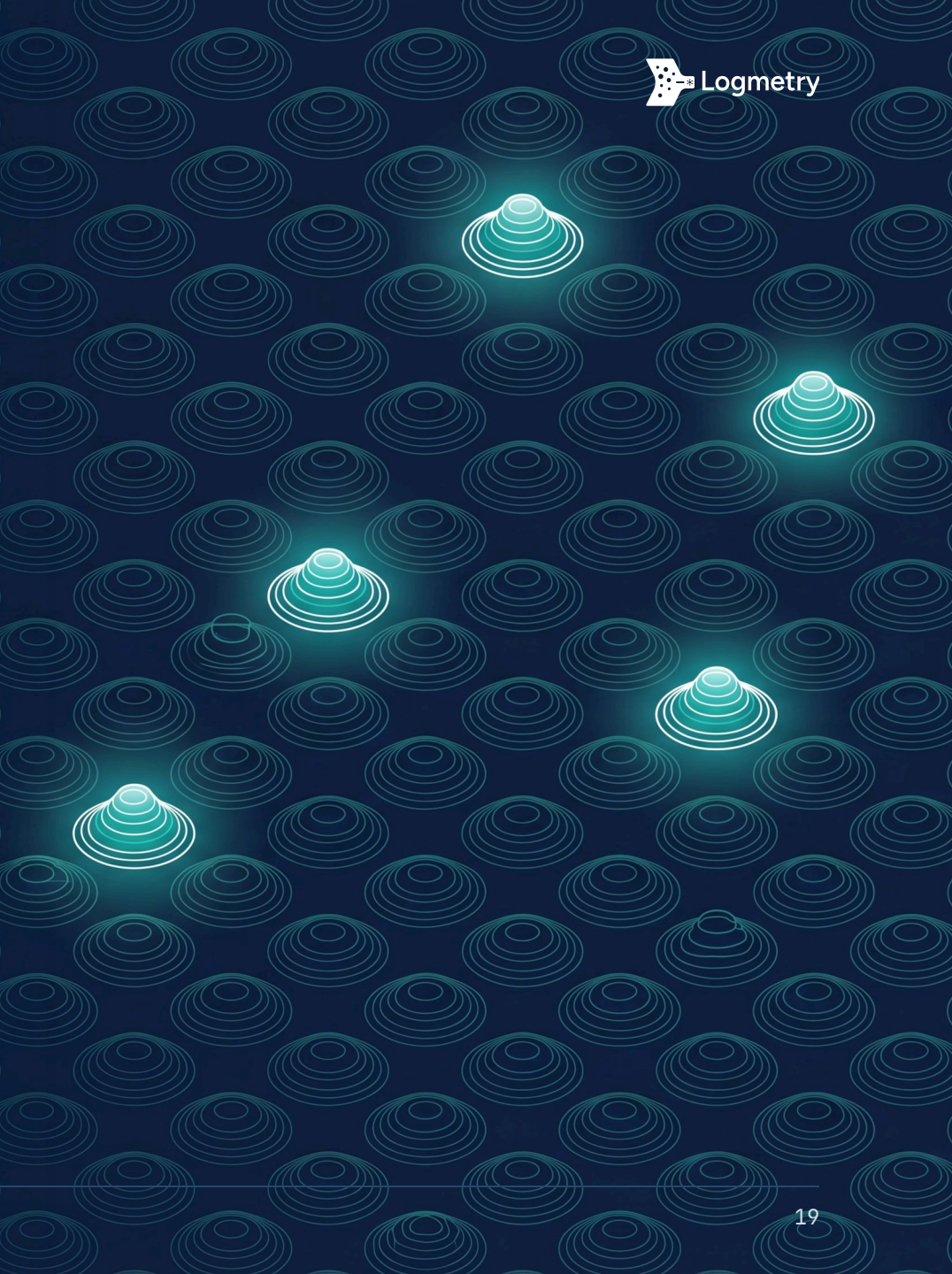
What it can fix. Pipeline routes, parsers, detection rules, alert thresholds, dashboards and the infrastructure under them, all held as code.

04

Footprint

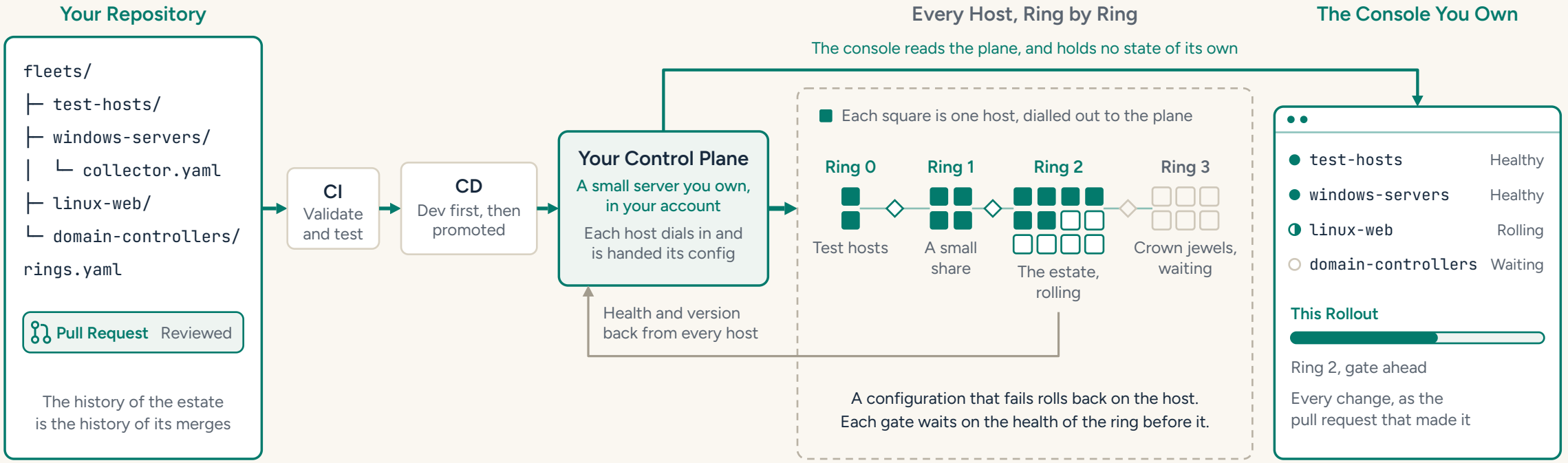
Collect once, then shrink the expensive tools to what earns its place.

Comes last. OpenTelemetry collectors on every host, nobody licenses them, under a fleet console in your account. Before a single host comes off per-node pricing, alert parity is proven tier by tier and in writing.



One open collector replaces every vendor agent, and every change to your fleet is a reviewed merge.

Exhibit 14 From a merged change to every host, ring by ring



The OpenTelemetry Collector is open source and carries no licence. Each host dials in over the Open Agent Management Protocol (OpAMP) to a control plane in your account and is handed its fleet's configuration, so the history of your estate is the history of its merges.

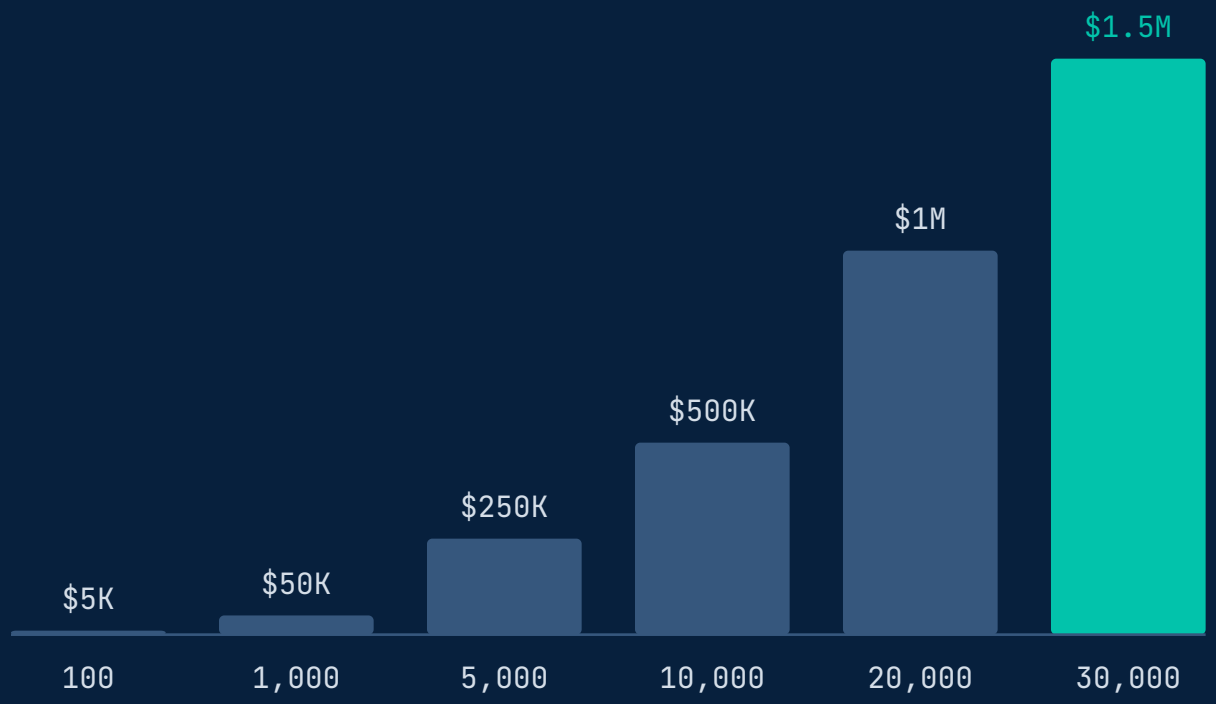
At fifty dollars a node, a 30,000-node estate pays $1.5M a year before anyone reads a dashboard.

$1.5M

Every year, at 30,000 nodes

Fifty dollars is roughly where a large estate lands after volume discounting, and published list pricing runs higher. Read it as the meter, not the saving. It is what the model charges, not a quote for what we take off the bill.

Exhibit 15 Per-node pricing at six estate sizes, drawn to scale



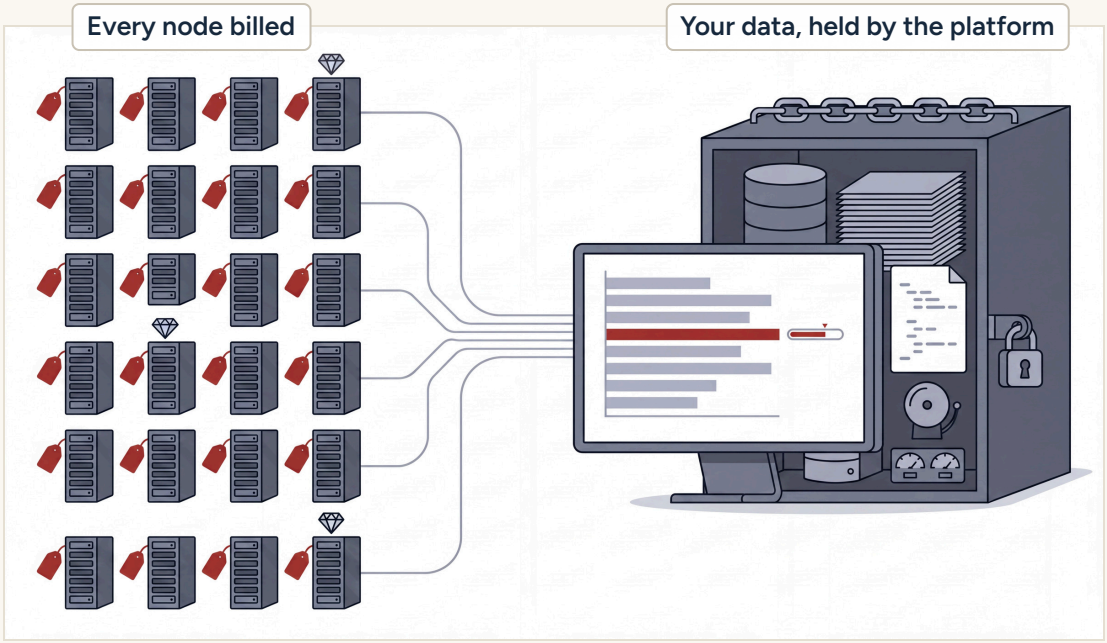
Nodes in the estate, and what the per-node model charges each year

Source: published per-node list pricing across the major infrastructure-monitoring platforms, and where a large estate lands after volume discounting.

Crown-jewel tracing keeps its per-host price. The ordinary nodes move to an engine you own.

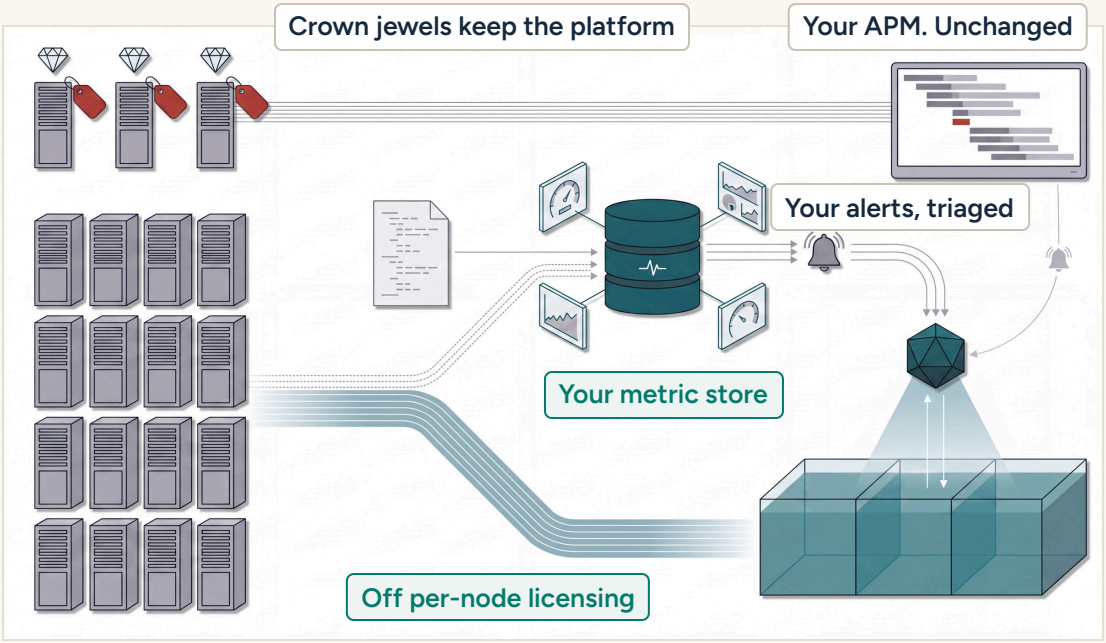
Exhibit 16 The same estate, today and after

Today



Service up, process running, CPU, memory and disk, watched by basic checks on every ordinary node. That tier is where the node count comes from, and so the bill.

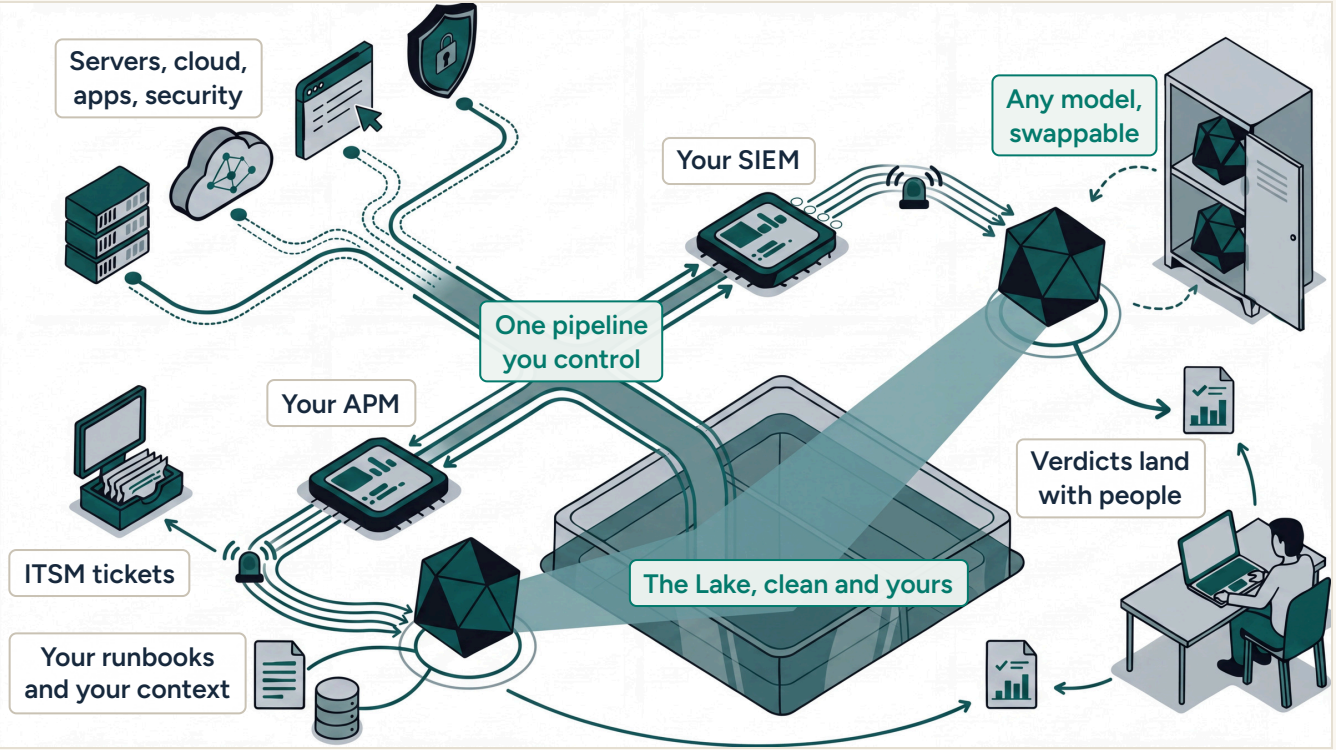
After



The same thresholds, durations and routing, exported through each platform's API as versioned code, on an open-source engine you own. Every rule is proven to behave as it does today before a host comes off per-node pricing.

The machine, complete: every agent works on ground you own.

Exhibit 17 Where the four phases land



Agents for Every Team

Security. A SOC agent works the SIEM alerts you choose against your Lake and runbooks, and appends the case to the ticket.

Infrastructure. A root-cause agent works the uptime alerts, and hunts the history for what nobody thought to alert on.

Engineering. Your developers hand requests and fixes to coding agents, and every one arrives as a pull request through the same CI/CD.

The business. Stakeholders ask in plain English and get answers on compliance, audit and cost, straight from the Lake.

> **AI native means an estate held as code.** Anyone can ask what is running and why. A change is an edit, the history is the audit trail, and a ticket becomes a pull request a person merges. A platform's assistant stops at the edge of its own console. This reaches all of it.

01 Foundation. A pipeline you control, in front of every platform. A clean Lake and a raw archive you own.

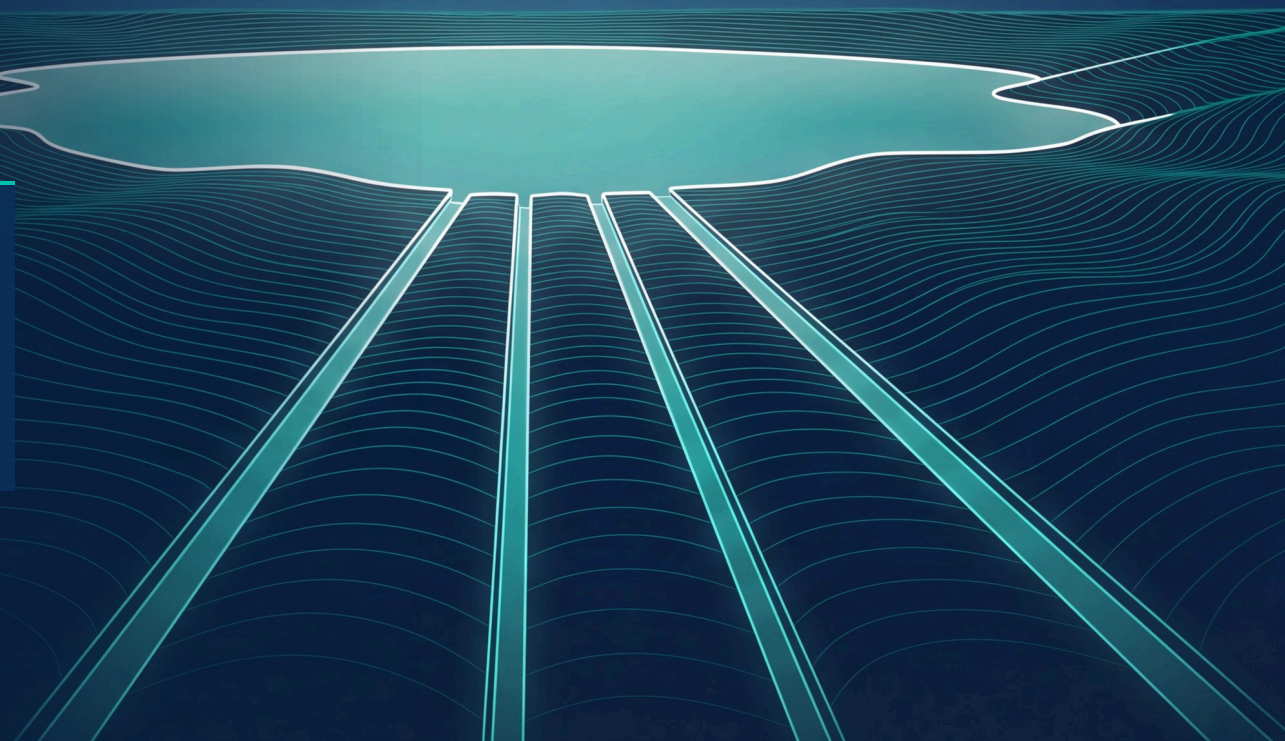
02 AI triage. SOC agents you own on the clean data: triage, root cause, threat hunting.

03 Remediation. Agents fix what they find, as pull requests a person approves.

04 Footprint. Collect once on an open standard, and the basic checks rebuilt as code.

From outside nothing moved. In between sits a layer you own.

Same sources, same destinations, same service desk. Less into your SIEM and budget back, clean telemetry your AI can read, and fixes that arrive enriched, as changes a person merges.



Start with the architecture review.

Bring your architecture diagram and we draw your version of this on your stack. Step one is the pipeline in front of your SIEM, proved on your data. No system access, no obligation.

info@logmetry.io
logmetry.io/contact