



The Death of the Traditional Search

How the security industry spent twenty years paying to keep
87% of its logs unreachable — and what finally changes that

MAY 2026

SCANNER.DEV

Table of contents

01	The tradeoff we normalized	3–7
02	Why every attempt to fix this has failed	8–14
03	The framework already assumes your logs work	15–18
04	The architecture that changes the equation	19–23
05	Natural language, AI agents, and why the data layer is everything	24–28
06	What full-fidelity actually looks like	29–34
07	Six questions to ask any vendor before you commit	35–38
08	Back to fundamentals: the tradeoff is over	39–42

PART 01:

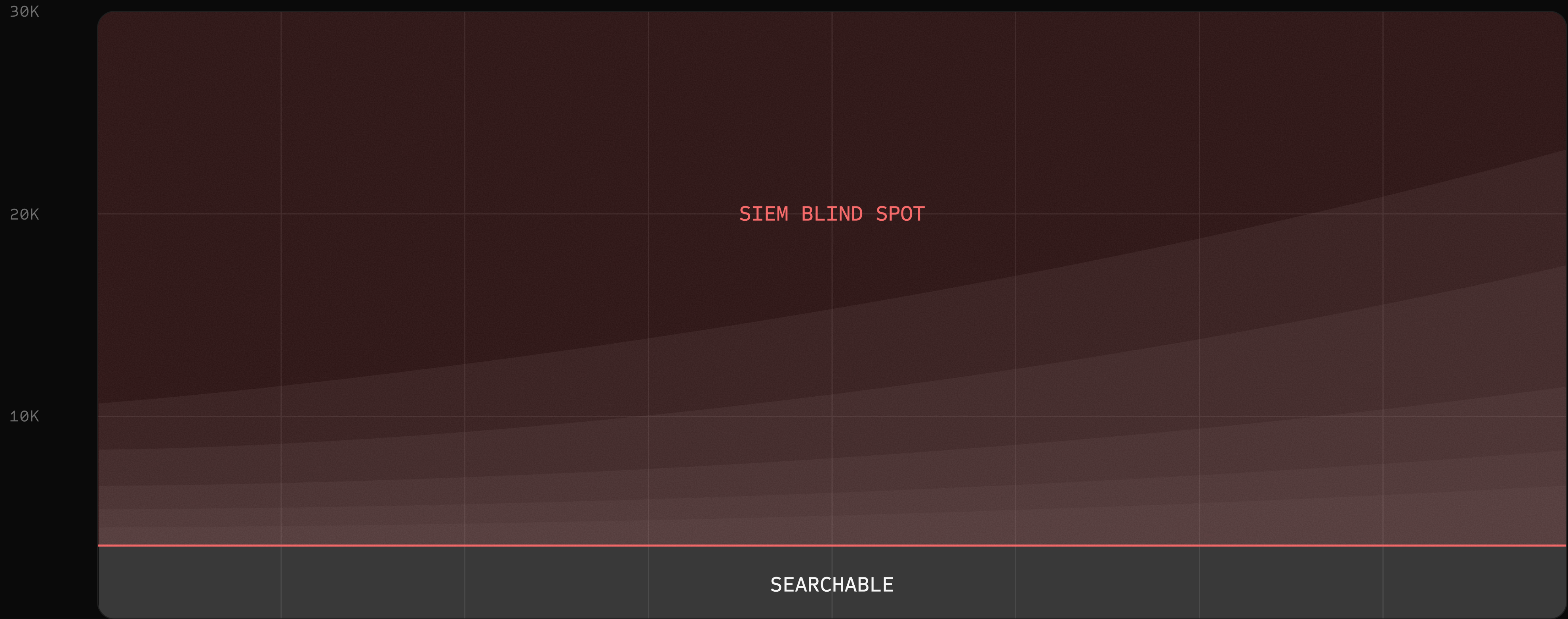
The tradeoff we normalized

Here is the situation as it actually stands, at companies that consider themselves reasonably mature in security operations: they generate terabytes of logs per day, store almost all of it, and can search a small fraction of it at any useful speed.

Mid-size enterprises: terabytes of logs daily. Large enterprises: upward of 50TB per day. Of that volume, among Scanner's users we see roughly 10 to 13 percent ending up in a searchable SIEM : Splunk, Datadog, Elastic, take your pick. The remaining 87 to 90 percent goes to cold storage because sending it to the SIEM was unaffordable.

BeyondTrust is a useful anchor. They generate 10TB per day. One terabyte of that goes into Elastic SIEM at a steep cost. The other nine terabytes sit in S3, where a query takes between 30 minutes and several hours and costs real money per scan. That is the median outcome of a decade of SIEM economics, not a niche edge case.

The obvious question is why the data sitting in S3 stays dark. The platforms capable of querying it at scale — Databricks, Snowflake, Athena — exist and work well for what they were designed for. The mismatch is workload. Those tools were built for data engineering: structured schemas, batch pipelines, SQL. Security investigation tends to be messier, entities ephemeral, and faster-moving. An analyst chasing an indicator at 2am across eighteen months of raw, unstructured logs needs results in seconds, not a schema-mapping exercise and a cluster spin-up.



90%

of enterprise security logs go to S3, where they are cheap to store and nearly impossible to search at investigation speed.

9:1

BeyondTrust's ratio of dark logs to searchable. 1TB/day in Elastic. 9TB/day in S3, where queries take from 30 minutes to several hours.

Legacy SIEM

Storage and compute priced together.
Teams capped ingest and lived with gaps.

The Cold Storage Trap

Logs routed to object storage to escape the bill.
Storage got cheap, search got impossible.

The Agent Era

AI agents iterate in seconds.
They need a data layer that keeps up.
Most can't.

DAILY VOLUME BY SOURCE

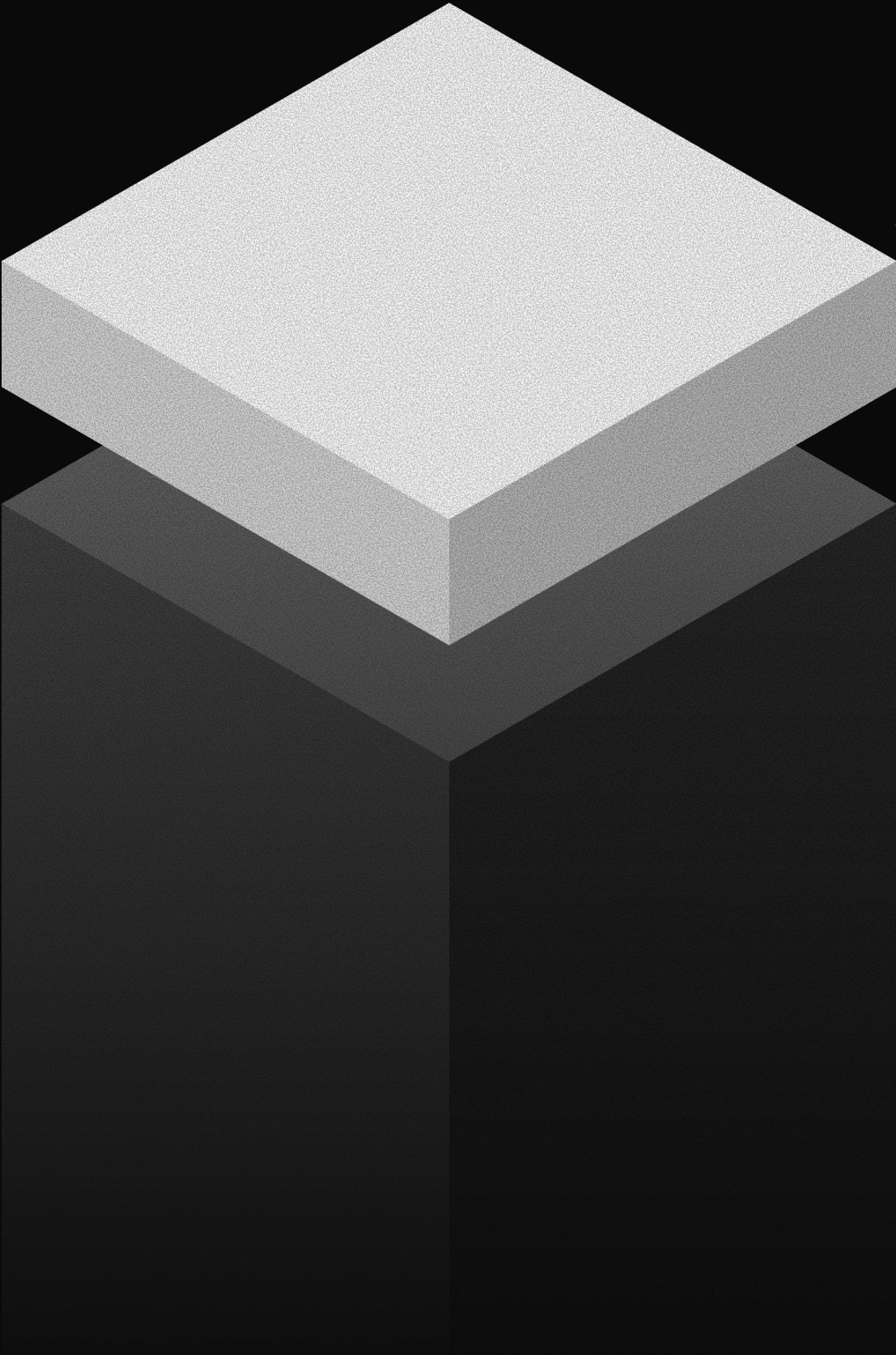
EDR + endpoint	37%
Cloud audit	27%
Network / VPC	19%
Identity (IdP)	11%
SaaS / app	6%

Compliance timelines have made this worse, not better.

Retention windows that used to run 30 days now stretch to a year in regulated environments. More data stored. The same 13% searchable. The gap between what security teams collect and what they can actually use has grown steadily wider.

The perversity here is that teams solved exactly the problem they thought they were solving. Storage is cheap and reliable. Every log source ships to S3. Compliance boxes get checked. The data exists. The problem is that existing and being usable are not the same thing, and the industry has spent twenty years treating them as if they were.

You already have the data. You are paying someone to make sure you cannot use most of it.



13% SIEM-INDEXED LOGS
Searchable at investigation speed

87% DARK DATA IN S3
Stored, not searchable, not usable

BeyondTrust:
7 TB / day sitting dark

“ ”

OMER SINGER

THE HIDDEN RISK OF
DARK SECURITY DATA

The security team turns to S3 and uses it as a dumping ground.

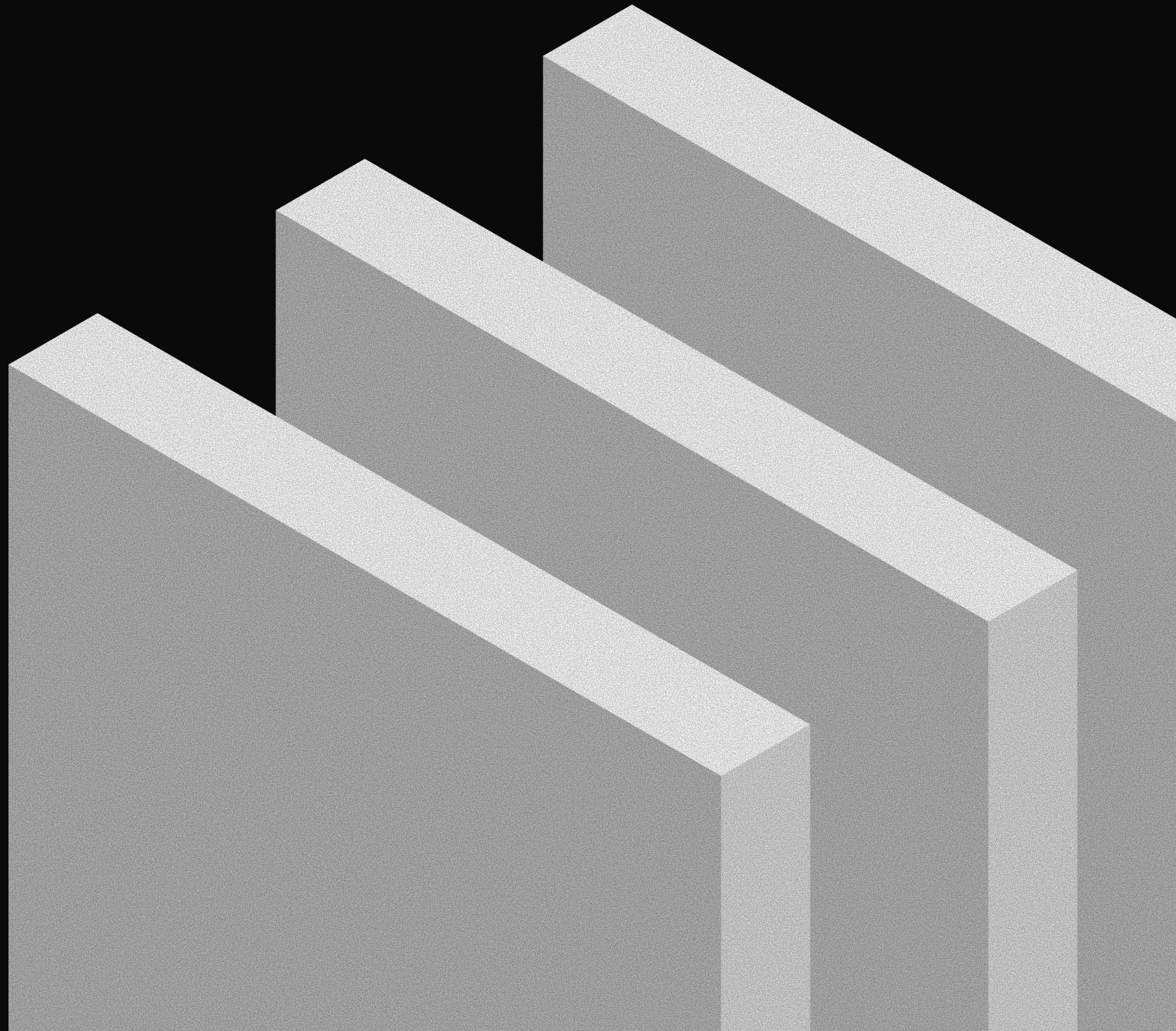
Over time, reliance on S3 buckets grows to a point where S3 becomes the organization's largest log repository.

PART 02:

Why every attempt to fix this has failed

The history of attempts to solve the SIEM cost problem is long enough to have its own periodization.

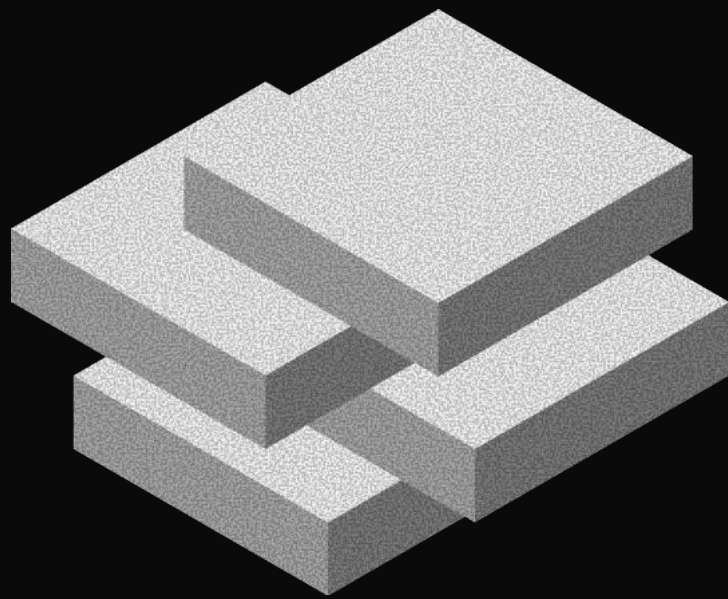
Three distinct approaches. Each one genuinely tried. None of them broke the underlying tradeoff.



APPROACH 1

DIY data lake

Hadoop → Databricks → Snowflake

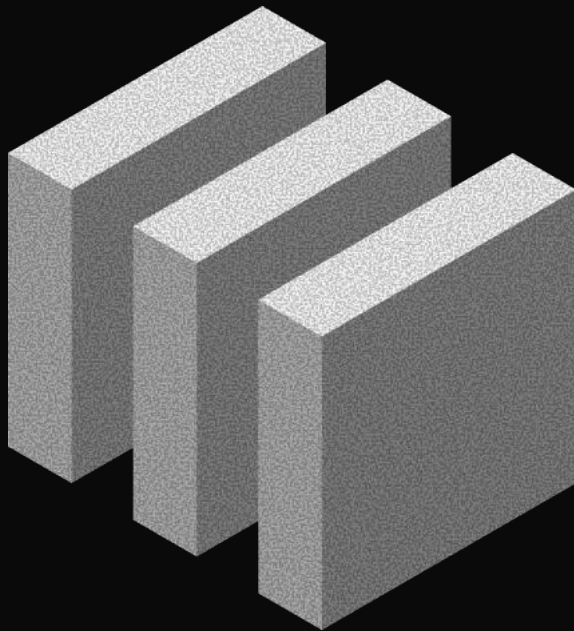


- × Unstructured, schema-inconsistent logs
- × Query speeds: 3–5 minutes
- × Analyst experience gap

APPROACH 2

Decoupled SIEM

Databricks + security layer

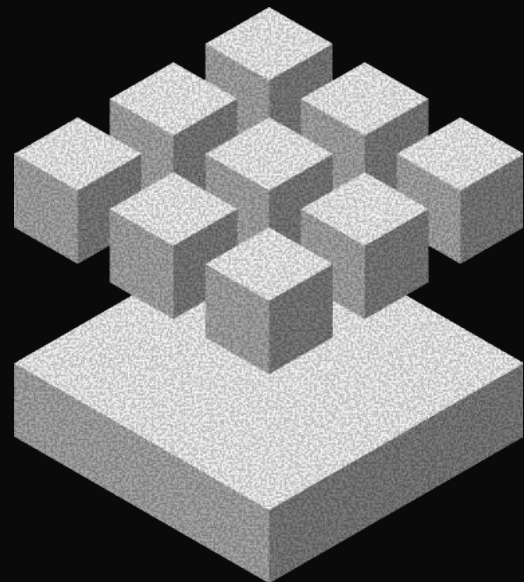


- × Multi-vendor blame loops
- × Databricks launched competing SIEM
- × Foundation still broken

APPROACH 3

Federated Search

Query via APIs in place



- × 2/3 of sources have no query API
- × Speed: minutes per query
- × “Our team would have quit.” — Benchling

Common thread across all three eras
None of them broke the original constraint between speed and cost. They worked around it.

APPROACH 1:

The DIY Data Lake

The data lake thesis was, at its core, correct: run your security analytics on the same general-purpose infrastructure your data team uses. Cheaper than Splunk at scale. More flexible. You own it.

Hadoop was first. Then Databricks. Then Snowflake. Each generation was genuinely more capable and more approachable than the last. Each still failed the security team for the same two reasons: complexity and speed.

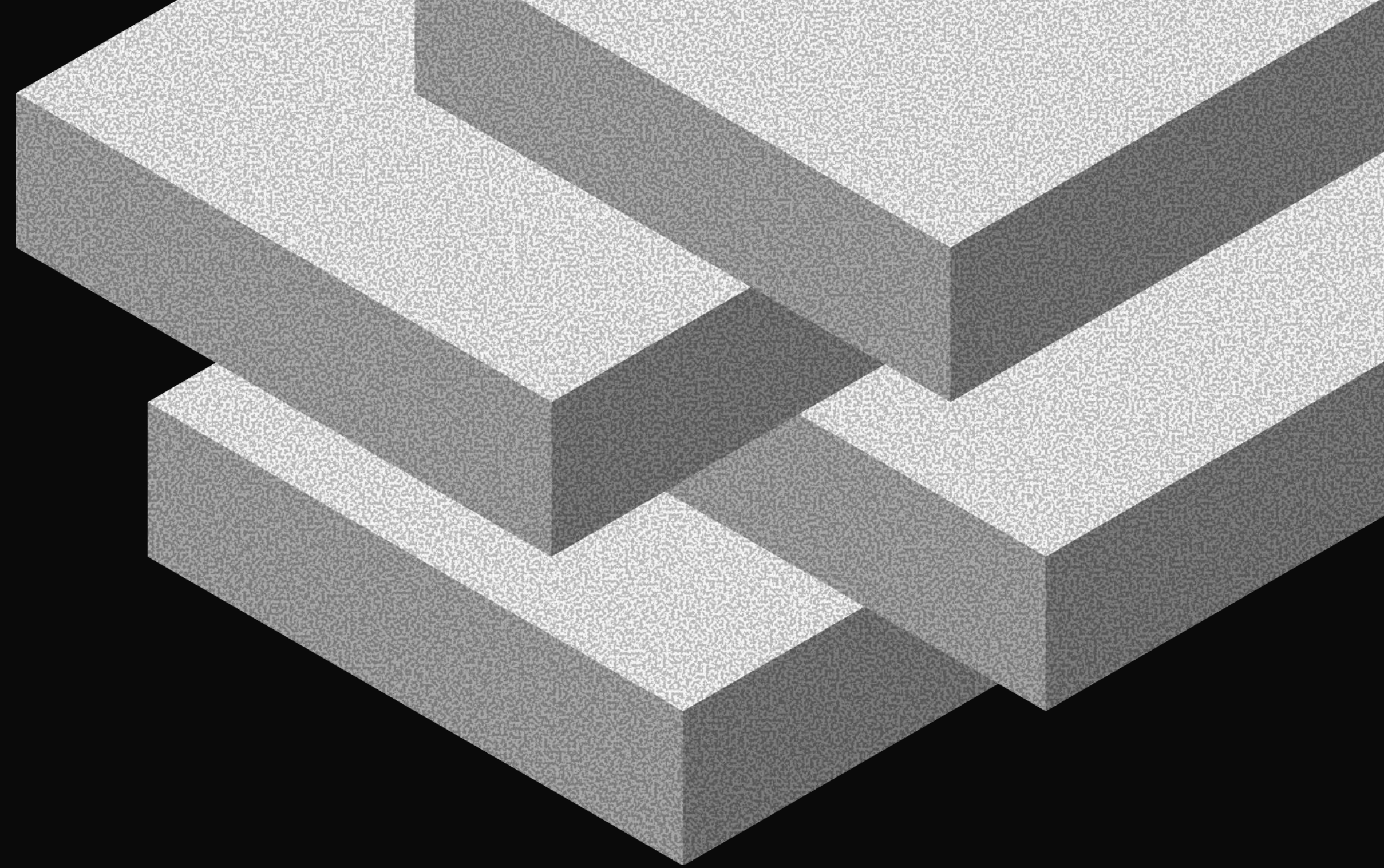
Logs are messy. They are unstructured, high-cardinality, schema-inconsistent across sources, and voluminous in ways that general-purpose analytics infrastructure does not optimize for. Getting them into a usable state requires parsers, data models, schema engineering, and ongoing maintenance work that most security teams do not have the bandwidth to sustain.

Several early adopters of the Splunk to Databricks migration found that cheaper infrastructure came with a hidden tax: losing on-the-fly field parsing and the ease of SPL, replaced with SQL queries that junior analysts cannot write and senior analysts do not want to write. Generative AI smoothed some of that friction but syntax fluency still determines whether detections get written and investigations get completed.

Speed compounds the problem. Snowflake's serverless model is cheaper than Splunk licensing at scale; that credit is real and should not be dismissed. But production deployments tell a

different story: test queries taking three to five minutes, warehouse startup latency requiring 16-minute lookback buffers just to account for delayed compute availability. Infrastructure designed for batch analytics was not designed for real-time investigations.

Former Gartner analyst Anton Chuvakin tracked DIY security data lake attempts from 2012 onward and revisited his assessment in 2022. His most optimistic conclusion after a decade of documented failures: "your security data lake project may still fail, but it probably does not have to." A lukewarm maybe, after ten years of trying, is not a success story.



APPROACH 2:

The Decoupled SIEM

The decoupled SIEM thesis: “keep your data in Databricks or Snowflake, add a security analytics layer engine on top” was a reasonable response to the DIY lake's complexity problem. Hand off the data infrastructure to people who specialize in it. Focus the security platform on detection logic and analyst experience.

The pitch was cheaper than a monolithic SIEM and more flexible. This reality introduced a new category of problems.

The cost math rarely survives contact with production. Teams that have run Databricks for non-ML workloads report bills landing at two to four times initial estimates, even after building in a 50 percent buffer. One practitioner summarized it well: "Databricks is a great tool, the problem is people forget it's just Spark. No one in their right mind should use managed Spark for 100GB of data." The platform earns its cost at genuine scale, hundreds of terabytes a day with heavy transformation and ML training. Most security teams that use it are paying for a data science platform to do log search.

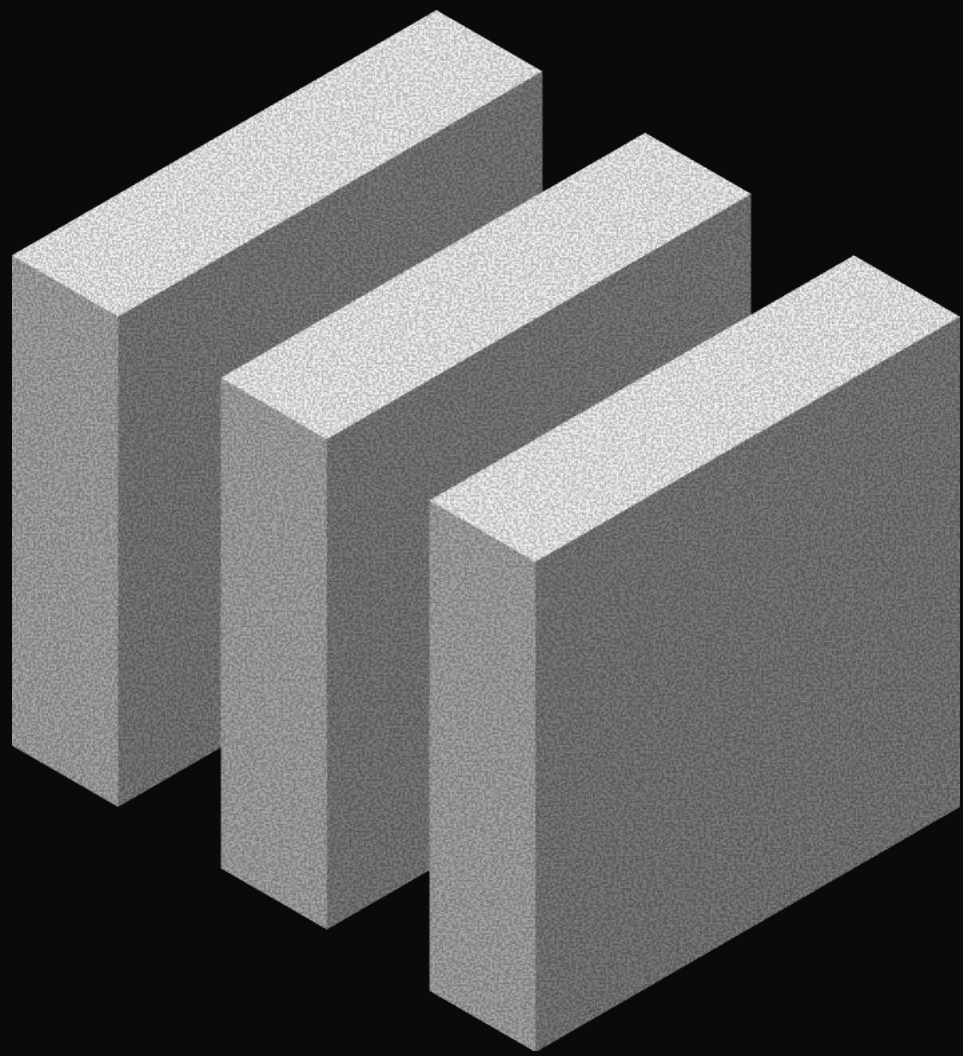
The developer experience compounds it. Databricks requires deploying to a cluster to test anything meaningful. Feedback loops that run 10 to 20 minutes per iteration are common. One practitioner put it plainly: it's managed Spark. Nobody should be running managed Spark on 100GB of data. The security analyst who needs to chase an indicator across six months of logs is not the user this platform was built for.

Multi-vendor complexity does not reduce total operational burden, it redistributes it across more seams. When something breaks in a

decoupled stack, you now have two vendors who have each other to point at. Ingestion lag? That is the data layer. Detection misfires? That is the analytics layer. Timestamp problems? Could be either. The security team sits in the middle of a blame-sharing exercise while the investigation stalls.

There is also a structural risk that rarely appears in vendor materials: your analytics vendor is now in competition with your infrastructure vendor. Databricks has launched its own SIEM. Snowflake has been building toward the same capability. The company you chose as your data foundation is actively building the product that displaces the security analytics layer sitting on top of it. Your decoupled stack is one product roadmap decision away from internal warfare between your two vendors over the same budget line.

And the underlying speed and complexity problems do not go away because you added a security layer. Databricks does not become analyst-friendly because Anvilogic is running on top of it. Hunters Security, another decoupled SIEM model approach, was even referred to as a SIEM without search. The query latency is still there. The warehouse startup delays are still there. You added a vendor contract and an integration surface.



APPROACH 3:

Federated Search

Federated search is the newest and most architecturally interesting alternative. The argument: stop centralizing logs altogether. Query data in place via APIs. No ETL pipelines, no batch jobs, no ingestion lag. Sounds elegant. In production, it encounters three hard limits.

Federated search is the newest and most architecturally interesting alternative. The argument: stop centralizing logs altogether. Query data in place via APIs. No ETL pipelines, no batch jobs, no ingestion lag. Sounds elegant. In production, it encounters three hard limits.

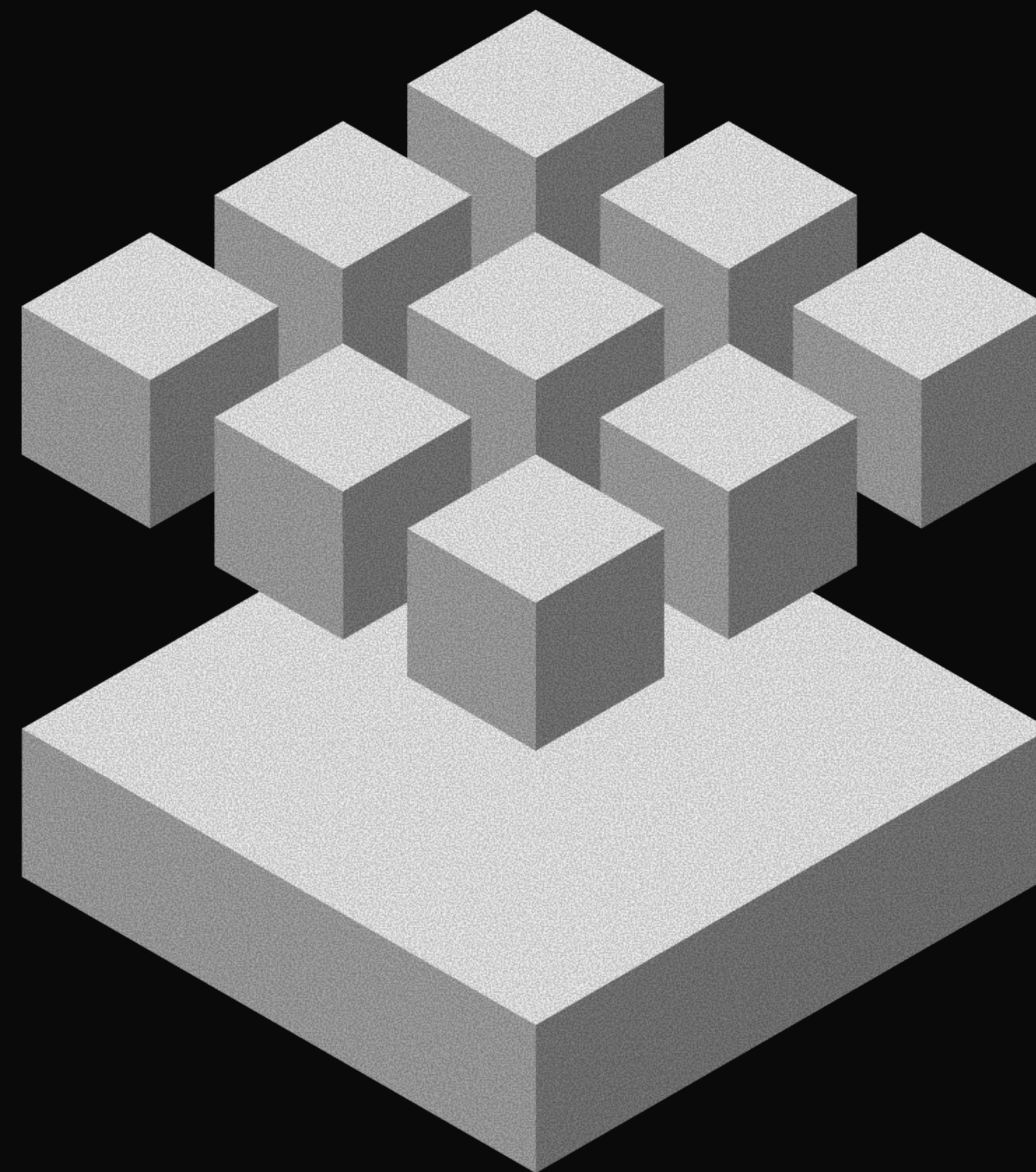
Coverage. Nearly every log source has native object storage export. Only about a third have APIs with the filtering and query capability required for incident response. Federated search's "100% visibility" claim holds only if every source has a production-grade queryable API. They do not.

Speed. Benchling tested federated search products directly against Scanner. Federated queries took minutes to tens of minutes. Their detection and response team was unambiguous: "Our team would have quit if they had to use Cribl Search. Way too slow." Real-time API queries and fast queries are not the same thing. Rate limits, API latency, and large

result sets all become bottlenecks simultaneously.

The ingestion lag argument, examined carefully, is a problem with batch ETL architecture — not with object storage. Federated search proponents correctly identify that traditional pipelines create 3 to 6 hour ingestion windows. True. But that is a Glue and Spark problem. Data flowing directly to object storage and getting indexed within one to two minutes has no such lag. The target they are shooting at does not apply to the architecture they are attacking.

Ramp ran into the coverage limit directly. It was not feasible to send Cloudflare DNS logs to Datadog because the volume made it cost-prohibitive. There was no API with adequate filtering. Federated search could not reach them. Scanner made them searchable in seconds. Phishing investigations that had been practically impossible became routine, on infrastructure already in place.



What all three failure modes have in common:
they tried to work around the tradeoff.

DIY lakes accepted slow queries.

Decoupled SIEMs accepted fragility.

Federated search accepted coverage gaps.

None of them broke the original constraint
between speed and cost. Scanner breaks it.

PART 03:

The framework
already assumes
your logs work

In 2025, former Gartner analyst Anton Chuvakin updated the SOC Visibility Triad, the decade-old framework that defined good security operations as logs plus endpoint plus network, and added a fourth pillar: application visibility.

The updated framework reflects how cloud-native environments actually work. SaaS platforms, service meshes, eBPF traces, and AI agents are all security-relevant actors that endpoint and network telemetry alone cannot see.

His framing for the updated importance of logs is worth sitting with: in cloud-native environments, logs have become more load-bearing. Not supplementary. Load-bearing. Not stored.

SOC Visibility Quad

Detection programs, AI agents, compliance — all resting on these four pillars.

The logs pillar is the most load-bearing in cloud-native ops — and 87% of it is hollow.

Adding a 4th pillar (application) makes this worse: even more log volume on the same broken foundation.

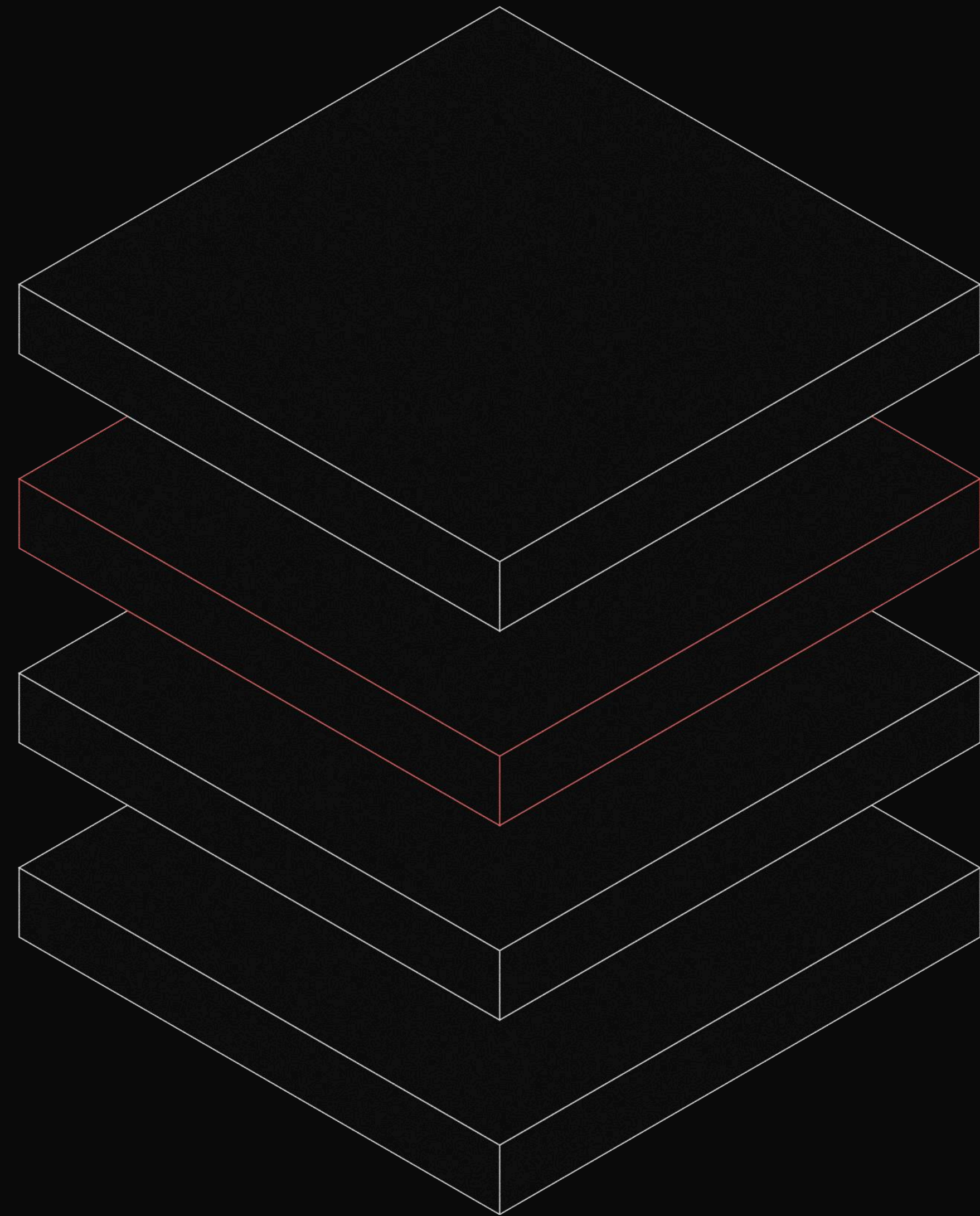
Endpoint EDR / XDR HOT - EDR alerts → SIEM (hot) - Threat detections - Quarantine events COLD - CrowdStrike FDR (full telemetry) - Process execution logs - File + registry events - Network connections SOLID PILLAR • PARTIALLY DARK	Network NDR / flow - IDS/IPS alerts → SIEM - Firewall deny events - Threat intel matches - VPC flow logs (full) - DNS query logs - Cloudflare / proxy logs - Palo Alto raw telemetry SOLID PILLAR • MOSTLY DARK	Logs ONLY ~13% SEARCHABLE SIEM + S3 - Auth events (sampled) - Cloud audit (partial) - Okta / IdP full logs - GitHub audit logs - CloudTrail full stream - Salesforce / SaaS logs MOST LOAD-BEARING • 87% HOLLOW	Application NEW eBPF / SaaS / AI Agents - AI agent activity logs - Service mesh traces - eBPF kernel traces - SaaS audit events - Slack / GitHub / Jira - LLM inference logs GROWING FAST • ENTIRELY DARK
---	--	---	---

The SOC Visibility Quad is the right framework. It is also built on an assumption that most security teams cannot meet: that the logs pillar is actually working.

Working meaning fully collected, indexed, searchable, and timestamp-accurate across all sources.

For most security teams, that assumption is false. The SIEM is 13% full. The other 87% went to object storage and sits there: retrievable only through queries that take 30 minutes and charge by bytes scanned.

The addition of application visibility as the fourth pillar makes this more acute, not less. Application layer telemetry: service mesh logs, SaaS audit events, generates high volume, unstructured log data by nature. The teams least equipped to handle full fidelity log volume at affordable cost are now being asked to add a fourth category of it.



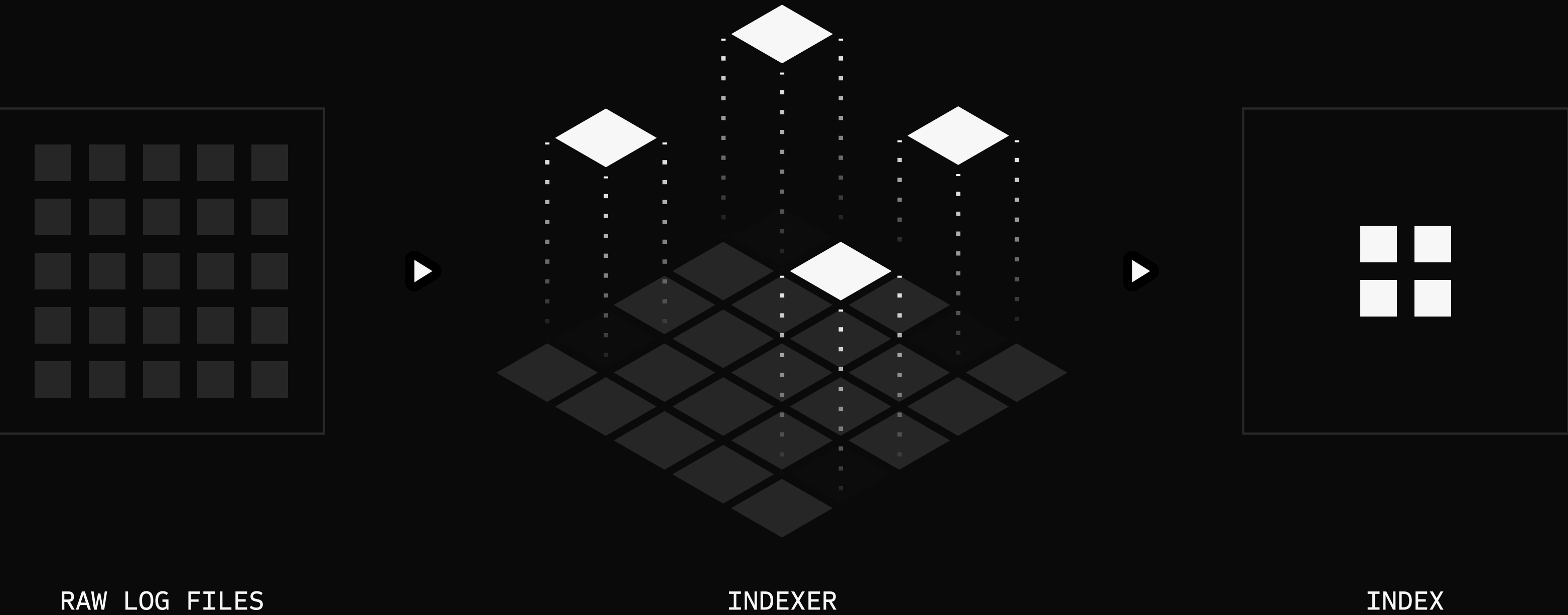
PART 04:

The architecture that changes the equation

Scanner is not a data lake in the sense any of the three eras previously attempted. It is not Hadoop. Not Databricks with a security layer. Not Athena with optimized query planning.

The architectural distinction matters because all three of those approaches share the same fundamental problem: they compute on data that was not indexed for text search.

Scanner builds an inverted index on top of object storage. Two components make this work in a way that prior approaches could not.



Inverted index

For instant search on object storage

When logs arrive in object storage, Scanner indexes them in place. No re-ingestion, no data movement, no ETL pipeline. The index narrows a petabyte to the exact set of matching log lines, then scans only that subset. This is not an architectural improvement over Athena. It is a different architecture entirely.

Athena scans columnar data. Scanner queries a text index. For security logs: which are predominantly text, high-cardinality, and searched by analysts using keyword and field queries the difference is not marginal.



Streaming query engine

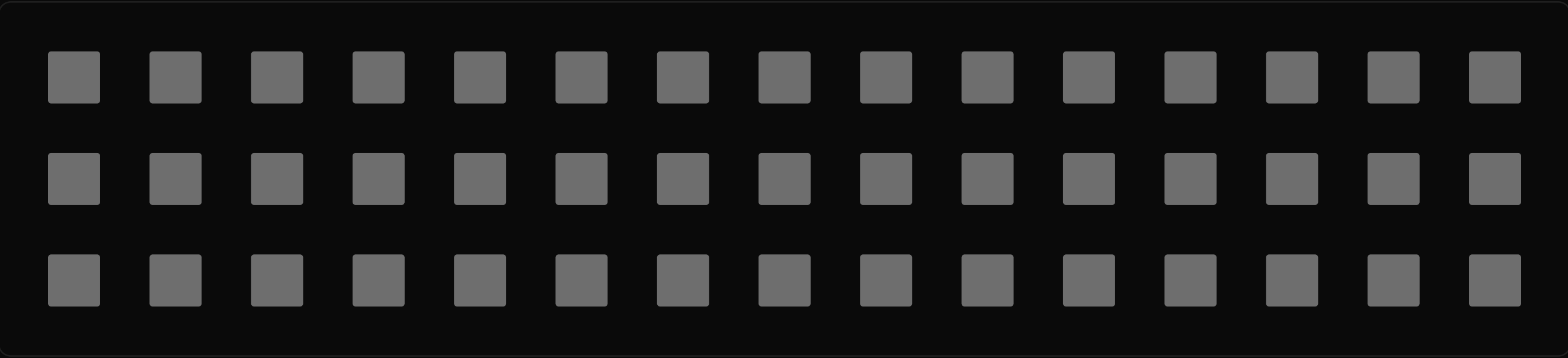
For real-time detections

Traditional detection rules rescan data repeatedly. Each new rule added to the stack costs more compute per log line processed. Scanner runs detections as data flows in, caching intermediate work across rules. Hundreds of complex detection rules run continuously without linear cost scaling.

Scanner also does not require centralization. Teams can have hundreds of buckets across different accounts, owned by different teams, managed under different governance frameworks. Scanner searches across all of them. The data stays where it is. The index enables what the raw buckets could not: instant, analyst-friendly search across the full data estate.

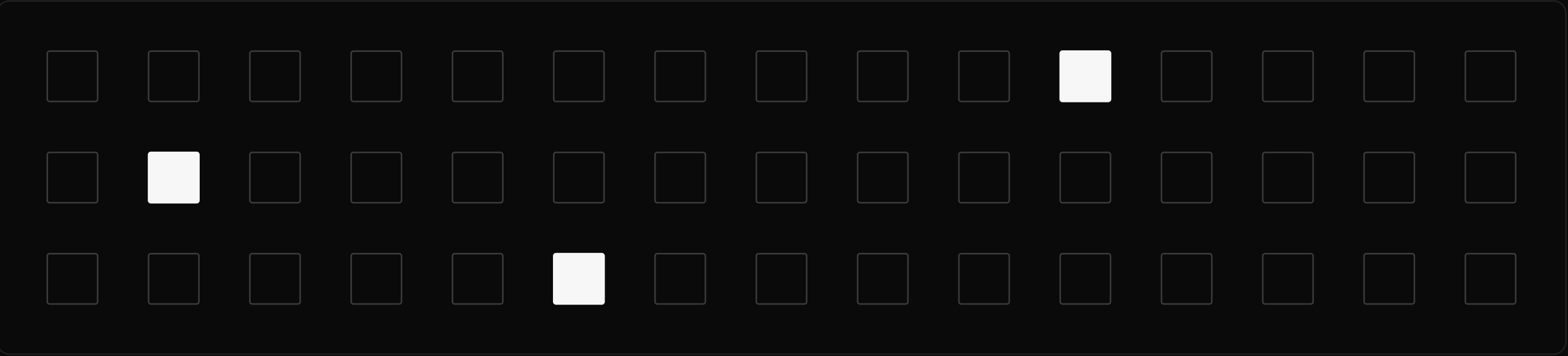
Athena

Full table scan	Reads entire dataset, regardless of match
\$75	Per query on 1 TB dataset
30 minutes	Query time, p99 latency



Scanner

Inverted index lookup	Only matching blocks retrieved via index
\$0.10	Per query, 750x cheaper
Seconds	Query time, 1TB/sec scan speed



BeyondTrust runs 10TB per day through Scanner at a highly economical rate. The same team was paying over 3x more in Elastic for only 1TB/day. Full-fidelity coverage at less than a third of the cost for seven times the data volume. That is a different kind of tradeoff.

PART 05:

Natural language, AI agents, and why the data layer is everything

The security industry is genuinely excited about natural language as the new interface for operations: writing detections in plain English, running investigations through AI agents that work continuously through the night.

The excitement is justified. The results depend entirely on what is underneath.

Look in my Scanner detection alerts and logs. Are people using root accounts in AWS? Any concerning activity there?

I'll help you investigate root account usage in your AWS environment using Scanner. Let me start by getting the context and then search for root account activity.

Get Scanner context

Now let me search for AWS root account usage. I'll query CloudTrail logs for root account activity:

Execute query

Good! I found root account activity. Let me fetch the detailed results:

Fetch query results

Summary: AWS Root Account Activity Analysis

Yes, there is **active root account usage** in your AWS environment. Here's what I found:

Activity Overview (Last 7 Days)

- 114 total root account events
- Activity concentrated on November 20, 2025 (19:57 - 20:37 UTC)

For two decades, query language was the bottleneck. SPL, KQL, SQL; mastery of the syntax sitting between an analyst and their data determined whether an investigation happened fast or slow, whether a detection got written or deferred. Agents dissolve that bottleneck.

When an agent can translate a plain English request into whatever query syntax the underlying system requires, fluency stops being the rate-limiting step. The competition shifts entirely to what the agent is sitting on top of. Speed of the data layer. Completeness. Timestamp accuracy. Availability across all sources, not just the 1% that made it into the SIEM.

The post-RSAC mood is a return to fundamentals. The appetite for AI SOC triage wrappers and alert explainers is fading. The question replacing it — are we actually using what we already have the right way? — is a data infrastructure question, not a query language question. When the interface layer becomes AI, the infrastructure layer becomes everything.

Scanner was built for the world where the interface layer is handled. The data layer is what remains.

Consider what speed means now that agents are doing the querying. When a human analyst ran a three-minute query, they could context-switch, grab coffee, come back. Agents don't. They iterate.

A security agent running an investigation executes dozens of sequential queries — authentication logs, process execution, IP pivots, adjacent hosts. Each is a step in a chain of reasoning. If each takes three minutes, a twenty-step investigation takes an hour, consuming compute, blocking the queue, and producing an answer that is now an hour stale.

The math compounds in the other direction too. The median time from initial access to credential theft is now measured in hours, not days. A threat actor who achieves initial access at 11pm and moves laterally at 2am has a three-hour window. An agentic system running on three-minute query infrastructure investigates that sequence with 60+ minutes of latency at the query layer alone, before alert triage, escalation, and human review.

Speed is now definitional. The gap between detecting and containing in minutes versus hours is not measured in analyst efficiency. It is measured in blast radius. Slow data infrastructure does not make your agents slower. It makes your breach larger.

Benchling proved this in production: federated queries took minutes. No agent runs a productive investigation loop at that speed. Agents also cannot build reliable timelines on timestamp-corrupted data. An AI that produces a confident sequence of events from misattributed timestamps is actively worse than no AI at all.

Scanner MCP launched in December 2025. Within weeks, six customers had built security agents on top of it: Notion, Ramp, Lemonade, Thirty Madison, Elise.ai, and Benchling. Notion's implementation is illustrative: an agent that investigates every security alert alongside CrowdStrike and Wiz, produces a written investigation summary with citations, and has the case 90 percent complete before any analyst opens their laptop on a high-severity 2am alert.

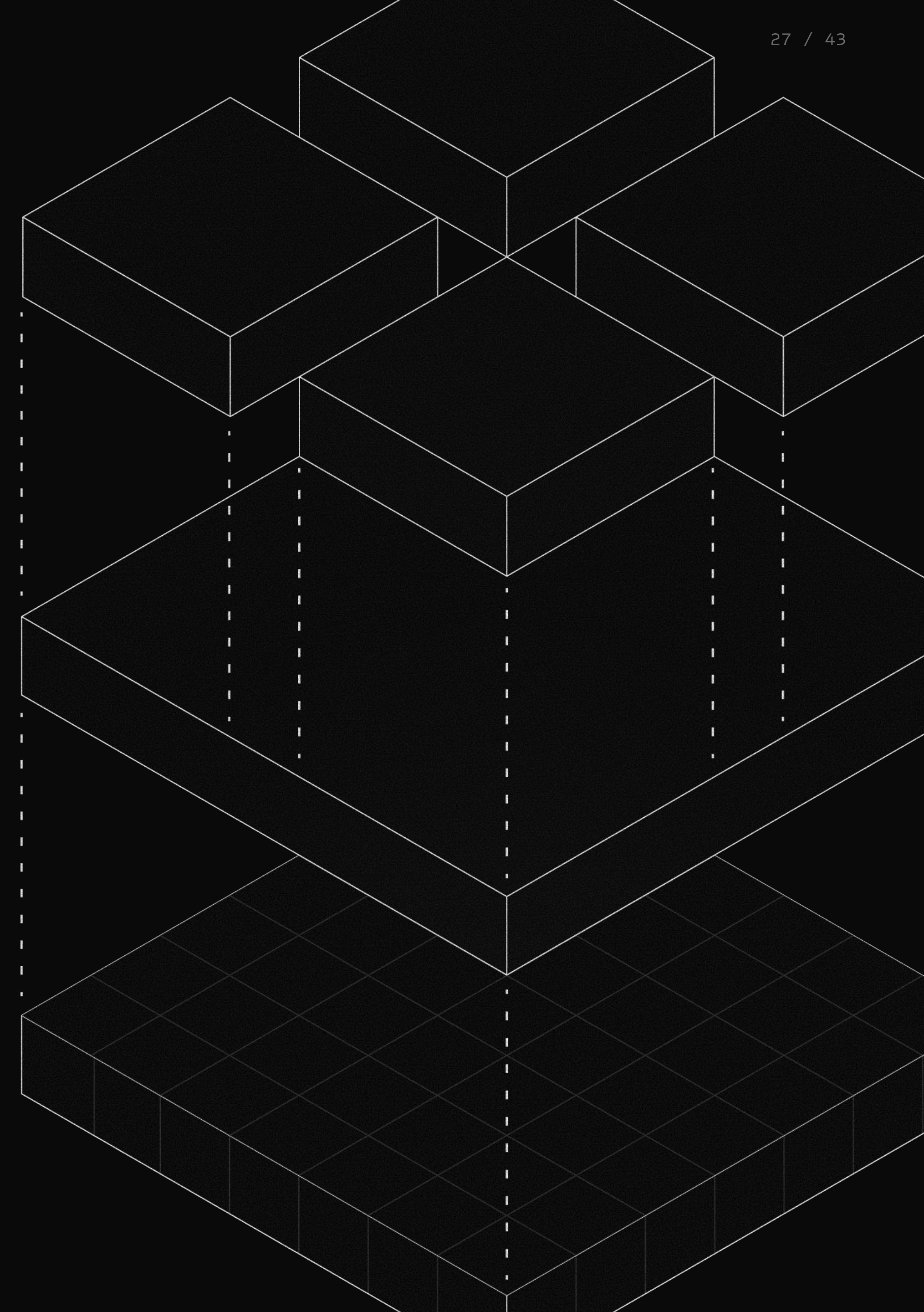
This works on Scanner because results come back in seconds. The agent asks many questions, tries multiple angles, iterates quickly — exactly how a skilled analyst works through an investigation. There is no agent-friendly version of a three-minute query.

Natural language detection is only as good as the data layer underneath it.

When the data layer is fast, full-fidelity, and timestamp-accurate, natural language becomes genuinely powerful.

When it is slow, partial, or timestamp-drifted, natural language is a better UI for the same broken experience.

Power lives in the data layer.
Everything above it is downstream.



There is no agent-friendly version of a 3-minute query. The architecture is the limit

Federated search agent loop 3-5 min per query		Scanner agent loop Seconds per query	
			
QUERY 1	Initial pivot Wait 3-5 min	QUERY 1	Pivot immediately to next angle Results in seconds
QUERY 2	Follow-up angle Wait 3-5 min	QUERY 2	Try multiple angles, iterate fast Results in seconds
QUERY 3	Agent stalls No productive investigation loop	QUERY 3-9+	Investigation narrative built, cited Seconds for each
2 queries in 10 minutes Investigation incomplete. “Our team would have quit.”		9+ queries in 10 minutes Case 90% resolved before analyst opens laptop (Notion, real production)	

PART 06:

What full-fidelity actually looks like

Abstract claims about full-fidelity security data are easy to make and make up the majority of SIEM and pipeline vendor websites. The production use cases that ground them are worth documenting.

Lemonade

Retention that works for auditors and analysts

Lemonade's previous SIEM kept logs for 7 to 30 days depending on the index. Getting older data meant rehydration fees, making it impractical for anything routine. Auditors ask about events from eight months ago. Investigators need to move fast when a breach notification references last quarter's activity. A 30-day window covers neither.

They moved to Scanner and kept 12 months of logs searchable. Audit requests and investigations now pull from the same data, at the same speed.

BeyondTrust

Detection without the ceiling

BeyondTrust generates 10TB per day. One terabyte per day was searchable in Elastic. The other nine sat in S3, queries taking 30 minutes to several hours. Scanner indexes the full volume at significantly lower cost. Same team, more data covered.

Detection works the same way. Most platforms have published limits that don't surface until after signing. Detection capacity is constrained by hardware or job pools: rules, dashboards, and analyst queries compete for the same ceiling. Per-run alert caps silently truncate during noisy incidents. Lookback restrictions force a tradeoff between depth and frequency.

Some platforms document these limits clearly. Others surface them during an incident. Scanner runs thousands of rules at one-minute cadence, sub-second even with multi-week lookback windows. No cap on rule count, concurrency, or alerts per run. When something noisy hits, the rules keep running.

ramp

How teams usually get here

Most teams don't start with a platform decision. They start with a compliance requirement. Logs need to go somewhere affordable for a year. Cold storage is the obvious answer. Scanner indexes what's already there. At some point someone runs a query and realizes it works. Then another log source gets added. Then another. The Datadog bill starts looking optional.

That is roughly how it went for Ramp. The data was already in S3. Cloudflare DNS log volume was too high to send to Datadog, making phishing investigations operationally impractical. Scanner made them searchable within hours of setup, with continuous detection running on top. No new infrastructure, no new pipeline. The decision to consolidate was obvious once search was no longer the constraint.

“ ”

JONATHAN JAFFE
CISO, LEMONADE

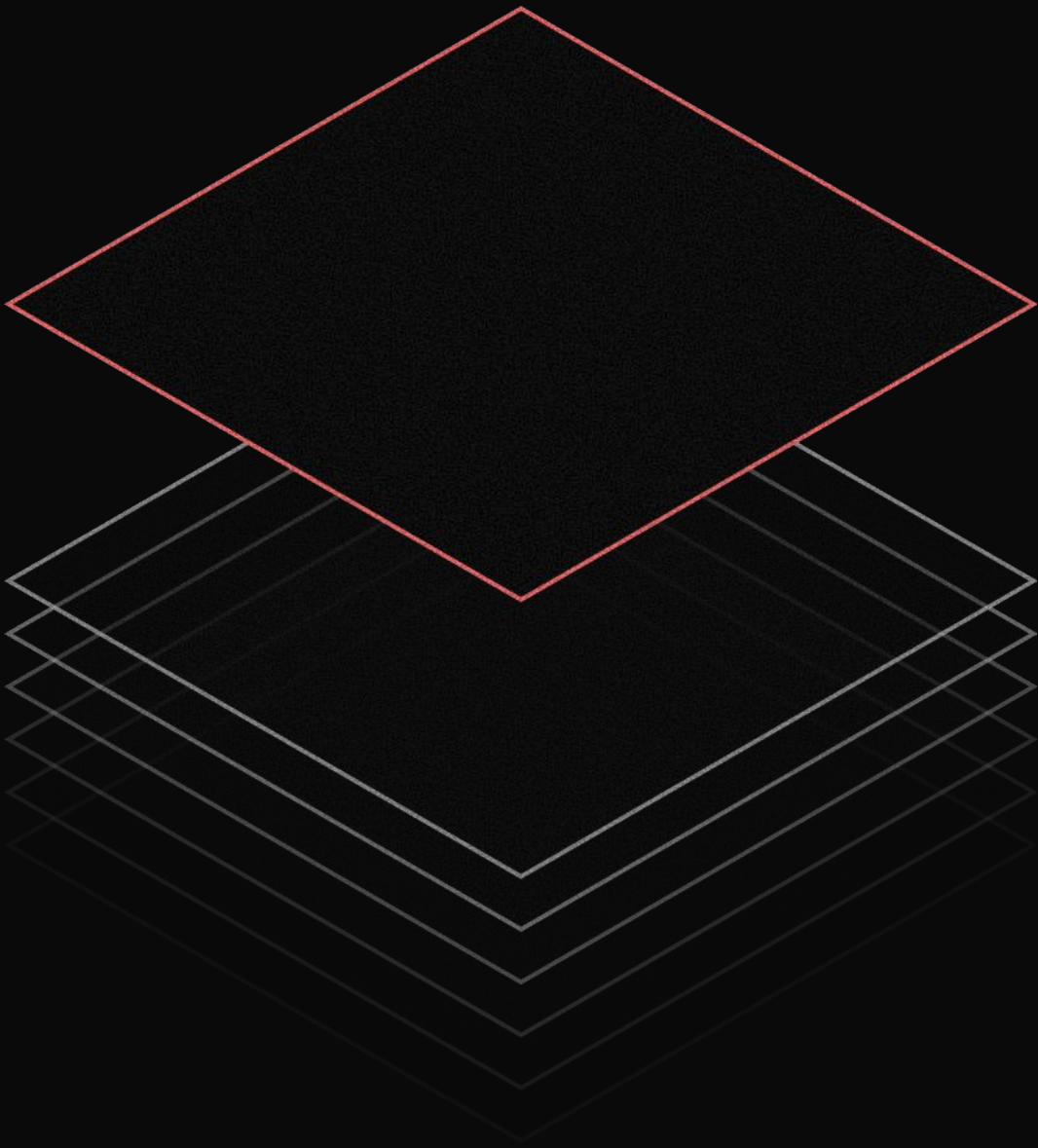
Scanner allows us to own investigations and be a go-to resource for other teams in the company.

The logging and searching is so fast, our team uses Scanner on a daily basis, not just for security tasks but for other purposes.

The Newer Entrants

Some newer entrants in the category take a different architectural path: building query layers on top of columnar databases that work well at small scale and get expensive quickly as volume grows. At a terabyte a day, the economics start to look familiar.

Others position as pipeline-plus-search, but the search half of that equation tends toward the slow end in practice, and the pricing tends toward the high end closer to what teams were trying to move away from than what they were hoping to find.



CUSTOMER IMPACT

BeyondTrust Detection engineering at scale	Ramp Phishing investigation at scale	Notion + Scanner MCP AI-driven alert investigation
BEFORE 1 TB / day - In Elastic — \$1m+ per year 7 TB/day sitting dark	PROBLEM Cloudflare DNS logs - Too expensive for Datadog - Logs already in S3, no API	BEFORE Agent does every alert - Alongside CrowdStrike + Wiz - Produces written summary with citation
AFTER 8 TB / day - Full fidelity — significant savings 8x data, ~70% cost reduction	OUTCOME Zero new infrastructure - Searchable in hours - Continuous detection enabled	AFTER Alert 90% resolved - High-severity alert at 2AM - Before analyst opens laptop
Hundred of continuous detection rules. No linear cost scaling per rule added.	Phishing investigations that were practically impossible became routine.	90% of queries on Scanner are now initiated by AI agents, not humans, across our customer base.

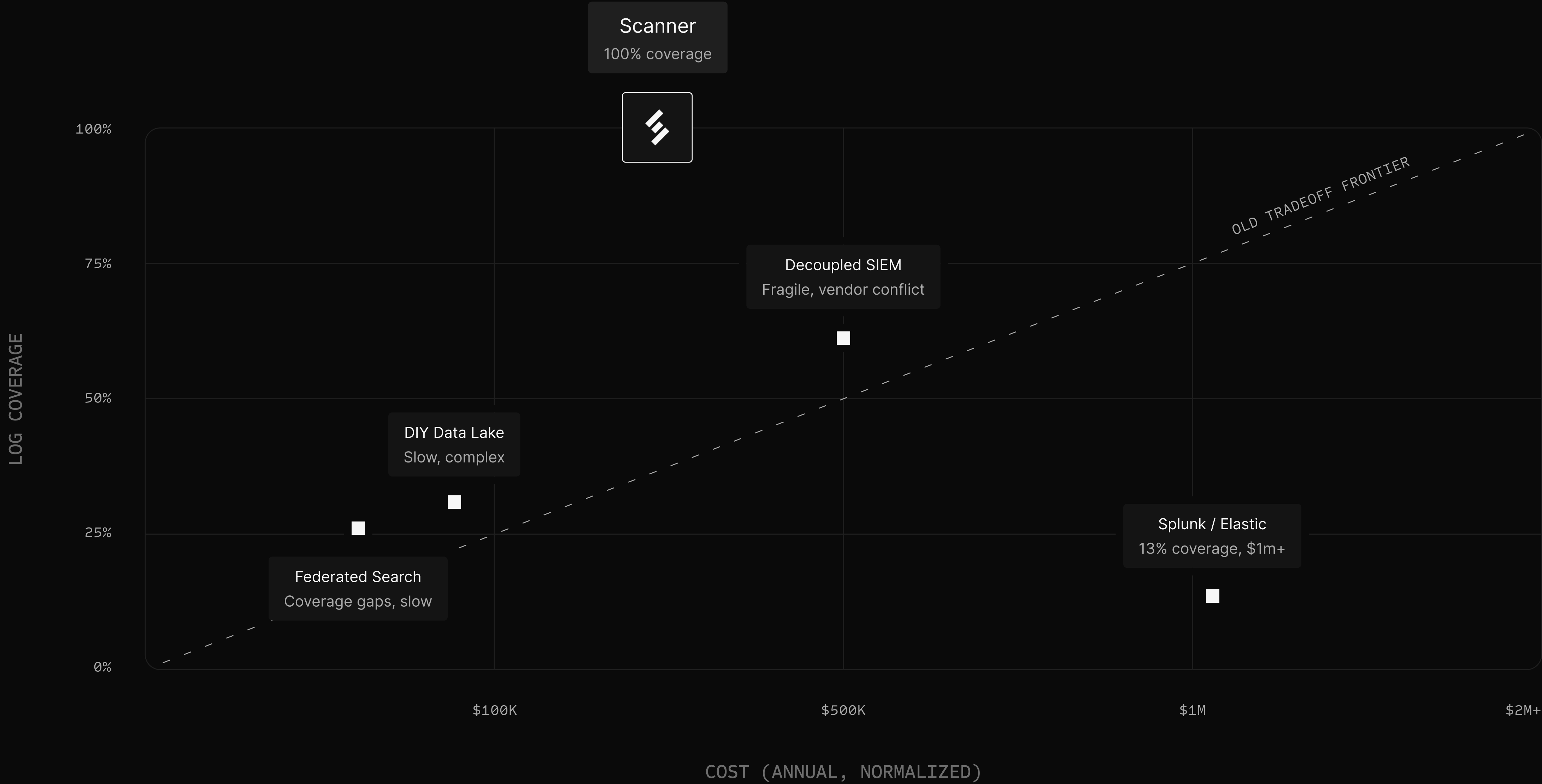
PART 07:

Six questions to ask any vendor before you commit

These questions are intended for evaluating any approach to the dark data problem — SIEM, federated search, decoupled model, or purpose-built index. Each one targets a specific failure mode documented in this deck.

Ask for specifics, not marketing claims.

QUESTION		FAILURE MODE IT TARGETS	WHAT A WEAK ANSWER SOUNDS LIKE
1	Coverage What % of log sources have real filtering APIs — not just streaming? Ask for a list, not a number.	Federated search Coverage gap — ⅔ of sources have no production query API.	"We support 200+ integrations." Ask which of your specific sources have query-capable APIs with real filtering.
2	Timestamp accuracy Which timestamp are you indexing — event time, ingest time, or pipeline time?	All approaches Corrupted investigation timelines — wrong sequence, wrong attribution.	"We ingest in real-time." Real-time arrival ≠ event-time accuracy. Azure overwrites timestamps before you ever see them.
3	Query performance What is p99 query time on 100 TB? 'Seconds' is not a number — ask for the benchmark methodology.	DIY lake 3-5 min query times. 16-min warehouse startup buffers.	"Sub-second on our demo dataset." Demo data is pre-warmed, pre-filtered, and not your data at your volume.
4	Vendor alignment What happens if your infra vendor launches a competing SIEM?	Decoupled SIEM Databricks launched a SIEM. Your data layer undercuts the analytics layer above it.	"We have a strong partnership." Databricks has roadmap decisions. So does Snowflake. Partnerships aren't contracts.
5	Cost at full fidelity What is per-GB cost at 100% of log volume — not just the 13% you currently send a SIEM?	All approaches Every SIEM looks affordable at 13% ingestion. The real cost hides in the other 87%.	"We're significantly cheaper than Splunk." Ask for a real quote at your actual daily volume. Then double it for retention.
6	Application visibility How do you handle eBPF traces, service mesh logs, SaaS audit events at volume without analyst experience collapsing?	DIY lake App-layer telemetry is high-volume, unstructured — the same infrastructure that failed logs fails harder here.	"We support OTEL." Supporting a format is not the same as making it searchable at investigation speed.



PART 08:

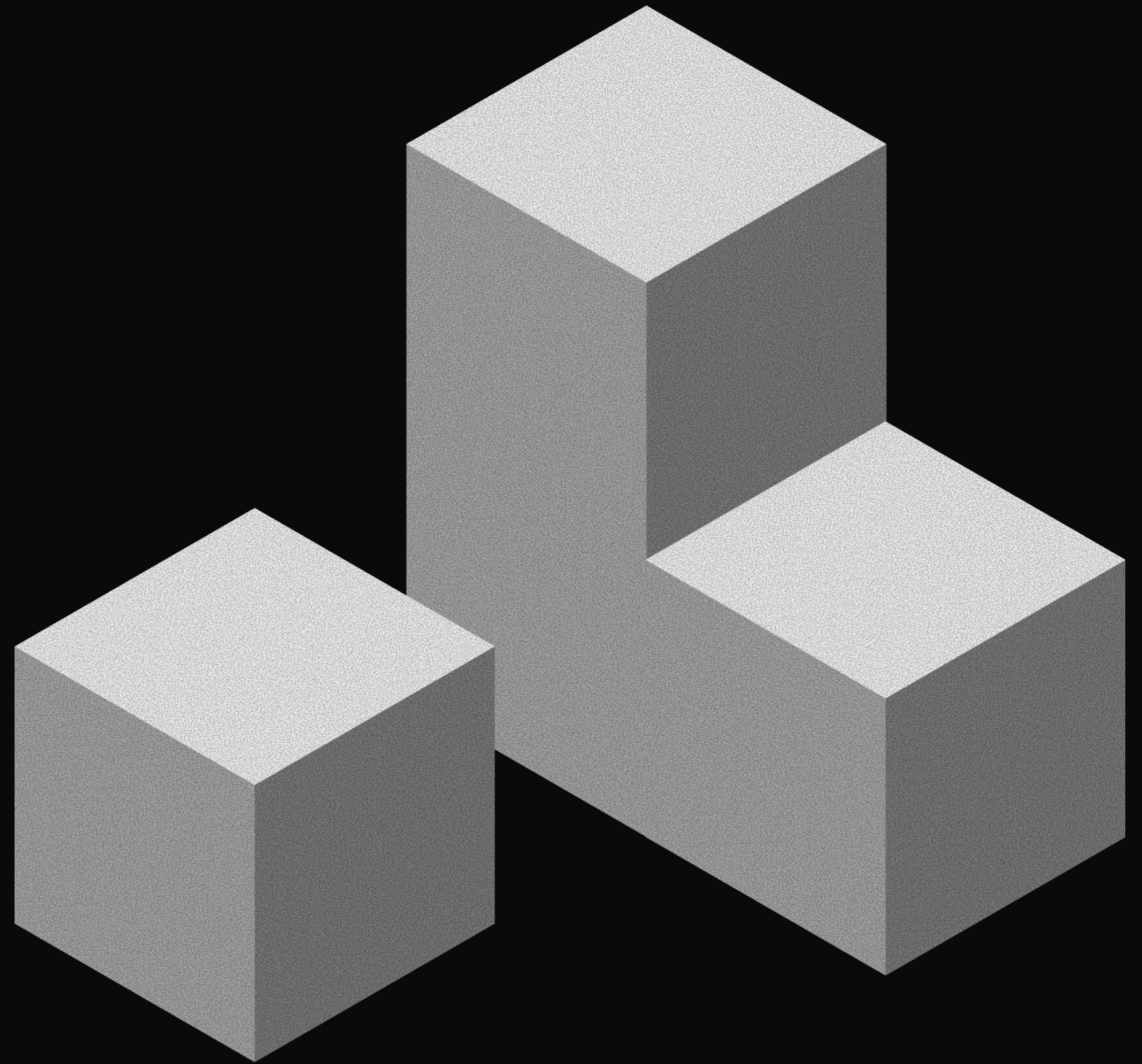
Back to fundamentals: the tradeoff is over

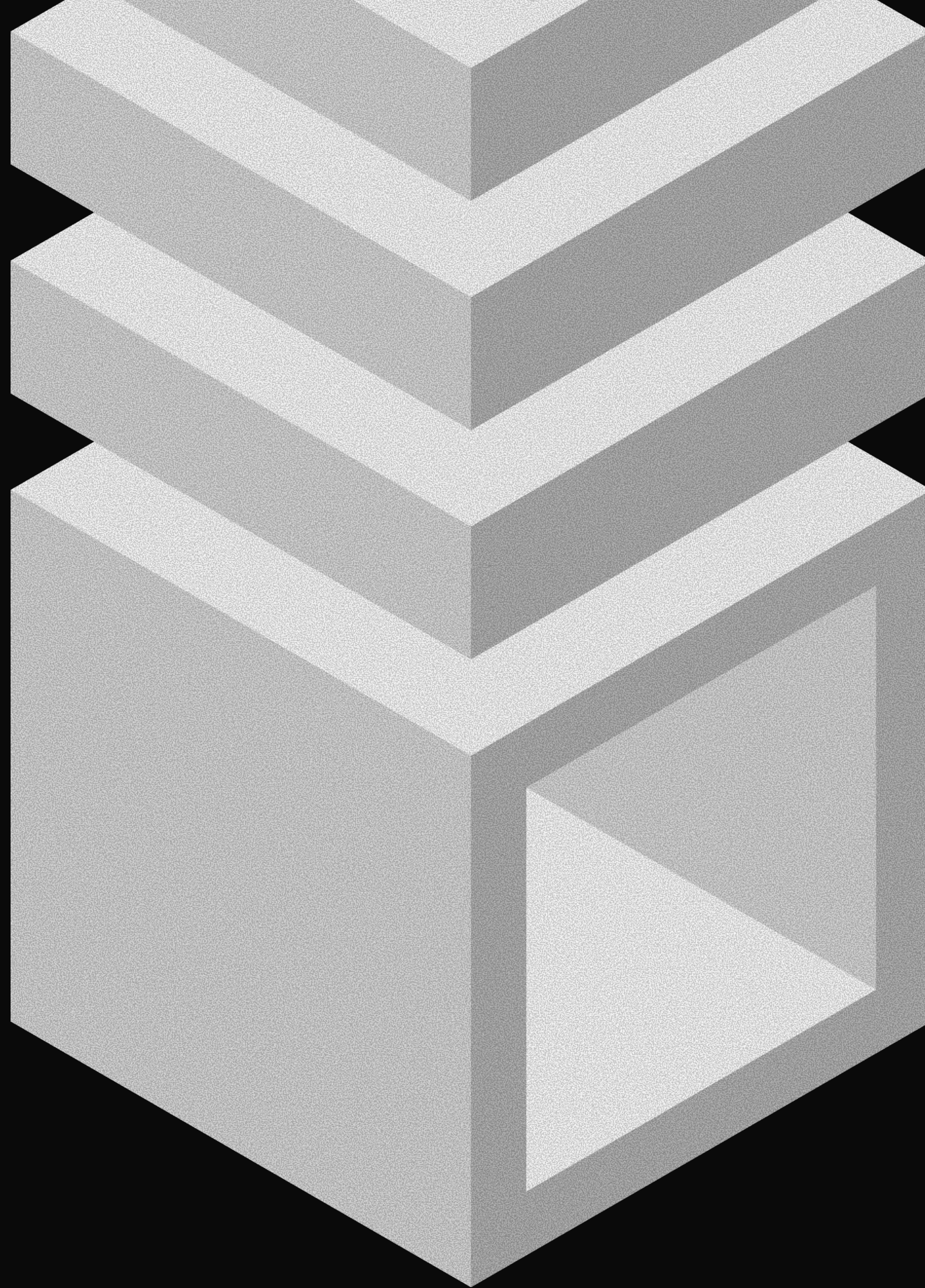
A dominant theme among security leaders after RSA 2026 is the return to the fundamentals.

Not new tools. Not new categories. The question that kept surfacing: are we actually using what we already have the right way? Asset visibility. Data you can actually access. Controls you can actually enforce.

A structural consequence of where the industry has arrived. When AI agents handle the interface, when the query language, the syntax, the SPL and KQL and SQL expertise that used to be the scarce resource are now abstracted away, the thing that actually determines outcomes is what's underneath. Speed. Completeness. Timestamp accuracy. Coverage of the full data estate, not just the affordable slice of it.

The "back to fundamentals" moment is the data layer moment. Every abstraction above it has been commoditized, or is being commoditized. The layer that hasn't been solved is the layer that was always load-bearing and was always being deferred.





That shift is Scanner's thesis, stated by the market.

The SOC Visibility Quad names logs as the most load-bearing pillar in modern cloud-native security operations. The industry has been building detection programs, AI agents, and compliance architectures on top of a pillar that is structurally hollow. 87% of logs are stored in a location that cannot support real-time search, continuous detection, or accurate investigation timelines.

The data is already in cold object storage. It has been going there for years, because it was the only place affordable enough to keep it. Every security team solved the storage problem. Nobody solved the search problem. DIY data lakes accepted the complexity and speed penalties that came with general-purpose analytics infrastructure. The decoupled SIEM model added vendors and operational seams without fixing the foundation underneath them. Federated search traded one set of coverage blind spots for another and required production-grade APIs that a lot of log sources do not have.

Scanner's answer is not a new architecture layered on top of old problems.

It is an inverted index built on the data that is already in object storage, returning results in seconds, with timestamps that reflect when events actually happened, at a cost that makes full-fidelity coverage economically rational for the first time.

The graveyard of failed data lake projects was not caused by bad intentions. It was caused by the wrong architecture applied to a real problem. The federated argument, that fast APIs can replace fast indexed search, has been tested in production, by practitioners, against real workloads. The practitioners voted.

Back to fundamentals means making the data you already have actually work. The question was never whether to store security logs. The question was whether to make them useful. That question now has an answer.

And "useful" has a new definition. In the era of agentic security operations, useful means fast enough to run twenty-step investigations in minutes, not hours. Fast enough that the agent's answer is still true when it arrives. Fast enough that dwell time shrinks instead of grows. The fundamentals resolve, precisely, to this: when the interface layer is AI and the threat actor is moving in hours, the only thing that determines outcomes is whether your data layer can keep up.

Scanner's answer to that question is seconds. The industry's answer, until now, was a spinner stuck on "Running."



A radically different way to detect threats in security data

[BOOK A DEMO](#)

[READ THE DOCS](#)

[SCANNER.DEV](#)

[LINKEDIN](#)

[TWITTER](#)