Fast and Cheap Observability

oodle.ai

Kiran Gollu,
Co-founder & CEO

What I will talk about today

1 | Key challenges with current tools

2 | How are we addressing them today?

3 | Key shifts defining the future

4 | A new first principles architectural approach that is faster AND cheaper

5 | What organizations need to do to stay ahead

Background

Kiran Gollu

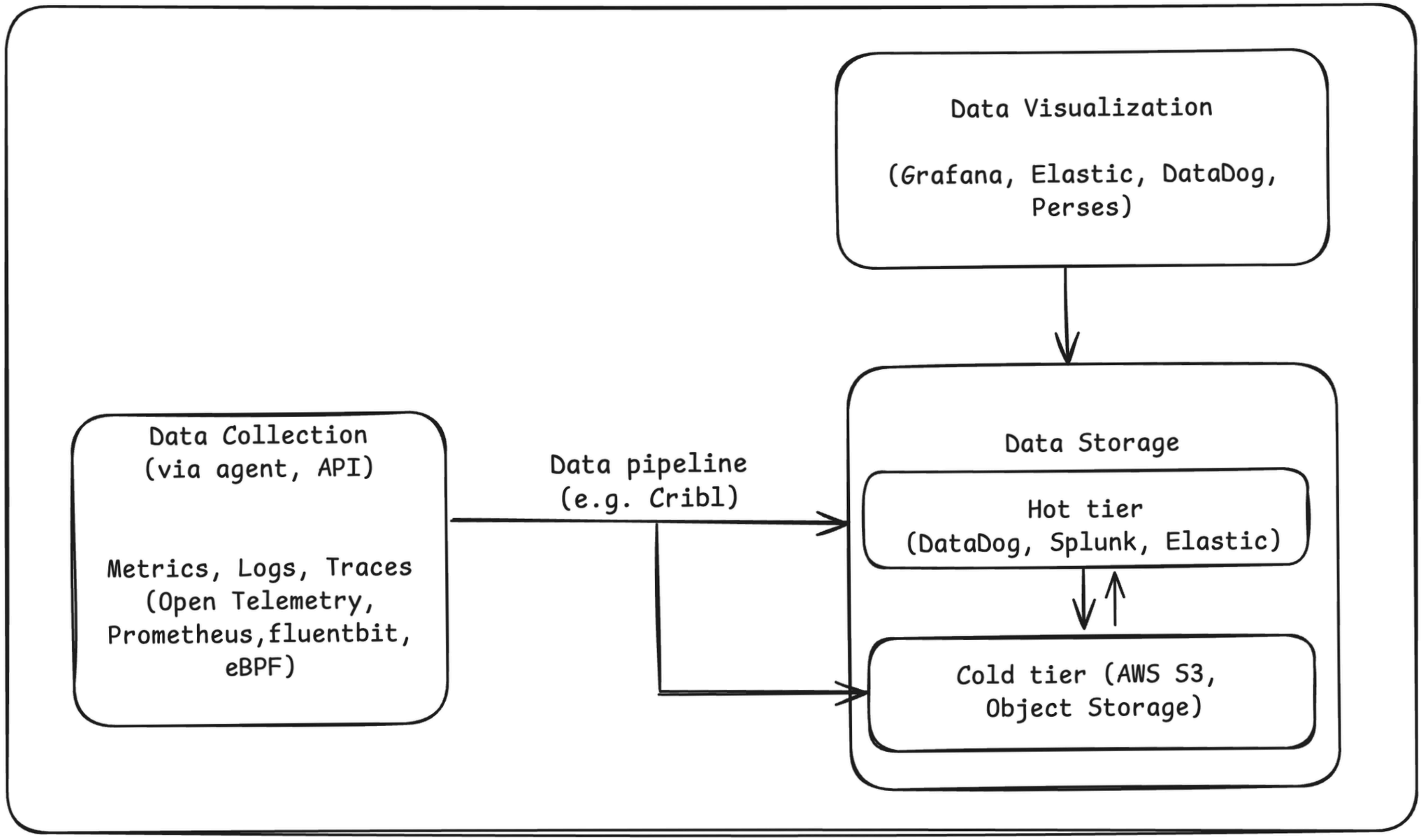
Co-founder/CEO	oodle.ai
Head of Cloud Platform Engineering (Scaling from 5M to 600M ARR)	rubrik
Founder, Incident Automation Startup	Neptune.io Y
Founding Engineer	aws | S3 DynamoDB

What is Observability?

1 | Understand what's happening inside a system by looking at its outputs—logs, metrics, and traces.

2 | It goes beyond basic monitoring, helping engineers quickly spot issues and figure out why they're happening.

Typical Observability Stack


Data Visualization
(Grafana, Elastic, DataDog, Perses)
Data Collection
(via agent, API)
Metrics, Logs, Traces
(Open Telemetry, Prometheus,fluentbit, eBPF)
Data pipeline
(e.g. Cribl)
Data Storage
Hot tier
(DataDog, Splunk, Elastic)
Cold tier (AWS S3, Object Storage)

Legacy Observability tools are broken!

Too Slow

Too Complicated

Too Expensive

What changed?

Switch to cloud native and AI

Explosion of data & micro-services

3rd line item in overall engineering spend after people and cloud, growing 2x faster

so what?

How does it impact various teams?

1 **App engineers don't care about breaking prod, i'm responsible, get woken up at 2am.**

2 **It's not reliable as we scale. it just doesn't work.**

"it's slow, queries time out all the time despite overprovisioning"

3 **Scaling our open source stack is hard**

"Rebalancing and upgrading elastic clusters is painful"

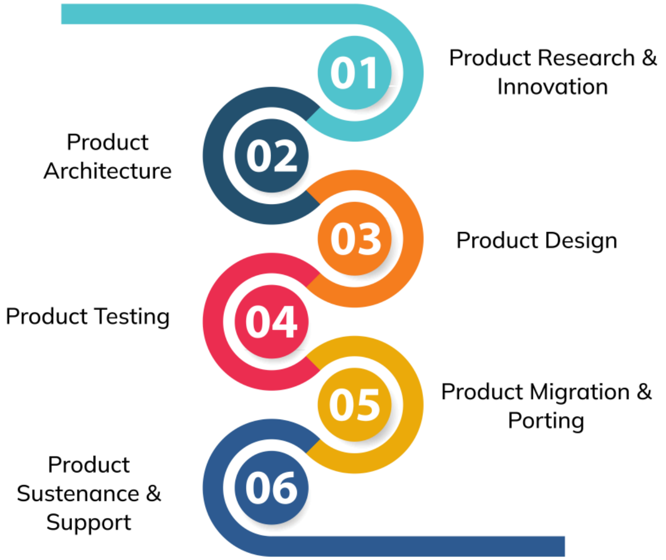
4 **Reduce observability to reduce costs**

"My boss asked me to reduce costs, but my product engineers don't like me anymore for blocking their metrics or removing logs"

Platform Engineering

1. Why is troubleshooting so hard? Too many dashboards, too many alerts.
2. I can't easily figure out what we send, what we use!
3. Why can't my observability system just answer any question at any cardinality fast?
4. Why can't app teams stop sending crazy scale logs and metrics, our customer experience sucks

Product Engineering



01 Product Research & Innovation

02 Product Architecture

03 Product Design

04 Product Testing

05 Product Migration & Porting

06 Product Sustenance & Support

1 My users notice problems before we do.

2 PMs constantly bug us with questions and interrupt our work. why can't they figure it out themselves?

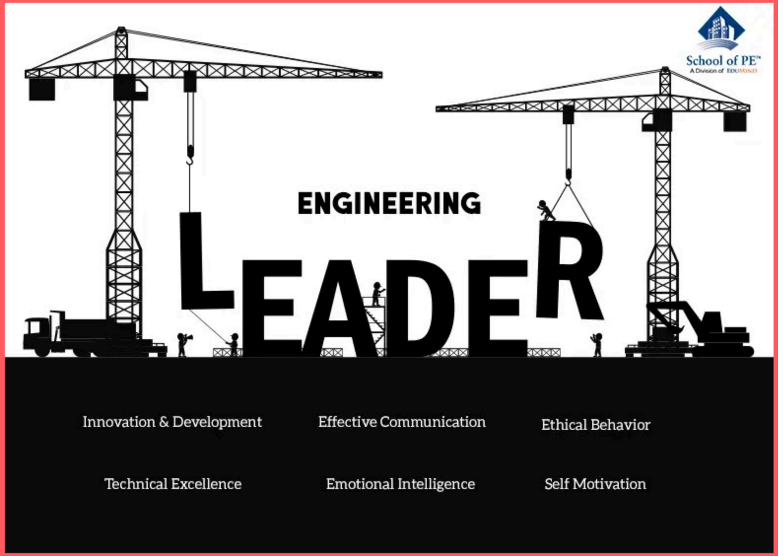
"are there any regressions due to latest release?"

3 Why can't I get answer to my questions asap?

"it's a crime to high cardinality data"

4 I want to get in and get out as fast as possible

"too many tools, it's complicated to debug - where do I start?"



1 We are heading to IPO. My CFO is asking me to reduce costs to improve our margins.

2 Commercial tools are expensive, scaling open source costs too much people time and CX challenges. What should I do?

3 We can't scale cost-effectively

4 Why should I build cost attribution features myself regardless of vendor of choice?

5 We are sick of migrations - weird lock ins, took painful 9+ months.

1. Using observability tools is not very intuitive at all. I've to constantly rely on my engineers for any questions.
2. Is there any regression in the latest release?
3. How do I measure product usage and adoption?
4. How do I measure the performance and availability for our premium customers? Are they happy?

okay it impacts multiple teams,
does any existing tools solve these problems?

Both Open Source and Commercial tools have challenges today!

	Too Slow	Too Complicated	Too Expensive
Open Source (Elastic, Prometheus, Grafana)	Not Reliable => OOMs, query timeouts	Not Horizontally Scalable Management Overhead Breaks at Scale (> 2M+ time series/hour)	Army of engineers & SREs :-) Over-provision CPU & Memory
Commercial (Datadog, Dynatrace, Splunk, Grafana Labs)	Too many dashboards! Gap: experienced vs new engineers Not optimized for E2E experience Each solves a piece of the puzzle	Balance budget and visibility Hard to do cost attribution Lock-ins: Migrations takes months/quarters!	3rd highest spend after people and cloud infra 20-25% of Cloud bill

How are we solving them today?

1st wave
Shift to Cloud

DATADOG

New Relic.

dynatrace

APPDYNAMICS

splunk>

2nd wave
Scalability and Cost Control/pipelines

Prometheus

Thanos

cortex

chronosphere

Cribl

Grafana Mimir

VICTORIA METRICS

Tiered storage helps, but not enough

Why legacy architectures are no longer enough for cloud and AI

1 | Not architected for cost-efficiency, instead for scalability and experience

2 | Row based data stores won’t work for high cardinality use cases

3 | Expensive & Slow: SSD disks (hot storage), object storage (long term backup)and replication adds more cost!

4 | Always running compute even when there are no queries on the system

5 | Based on lift and shift on-prem architectures

Why now?

1 | Observability has become a data problem

2 | Separation of Storage and Compute Phenomenon (Snowflake)

Innovations in data warehousing, DB, data streaming architectures

3 | Serverless Wave

Performance inprovements (Parallelism & cold start latency)

4 | Generative AI

But, observability data has different properties

1 | You mostly care about recent data

2 | Time series data is highly compressible

3 | > 95% of data you send is never queried

4 | Queries need to be fast (p99 latency < 3 seconds)

5 | Complex aggregations are required, yet speed is important!

What are the requirements of an ideal solution?

1. Observability is all about resolving incidents quickly, getting answers fast.

2. Architected for cost-efficency from Day 1

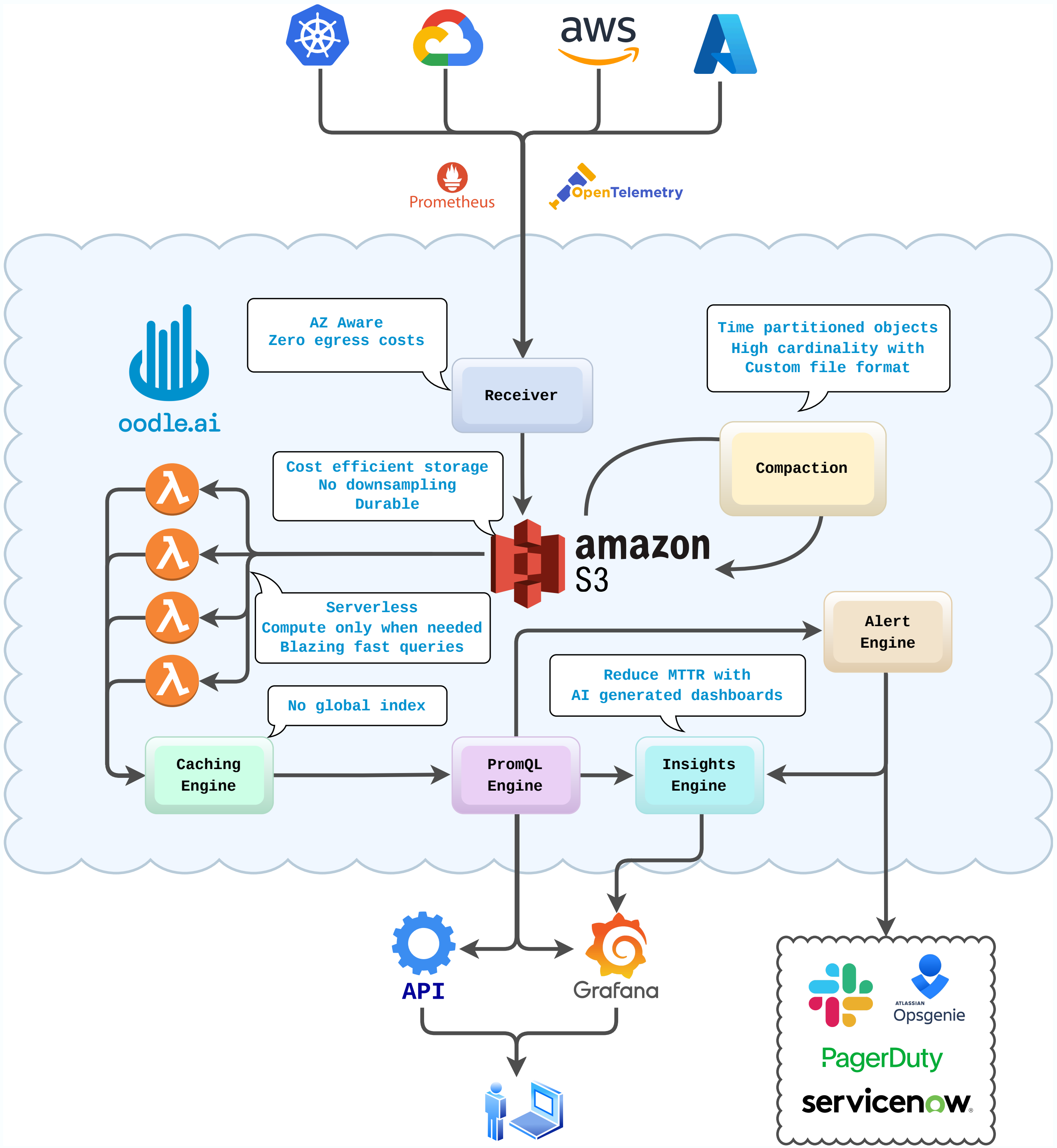
3. Simple, OSS compatible with no-lockin

A new first principles appraoch

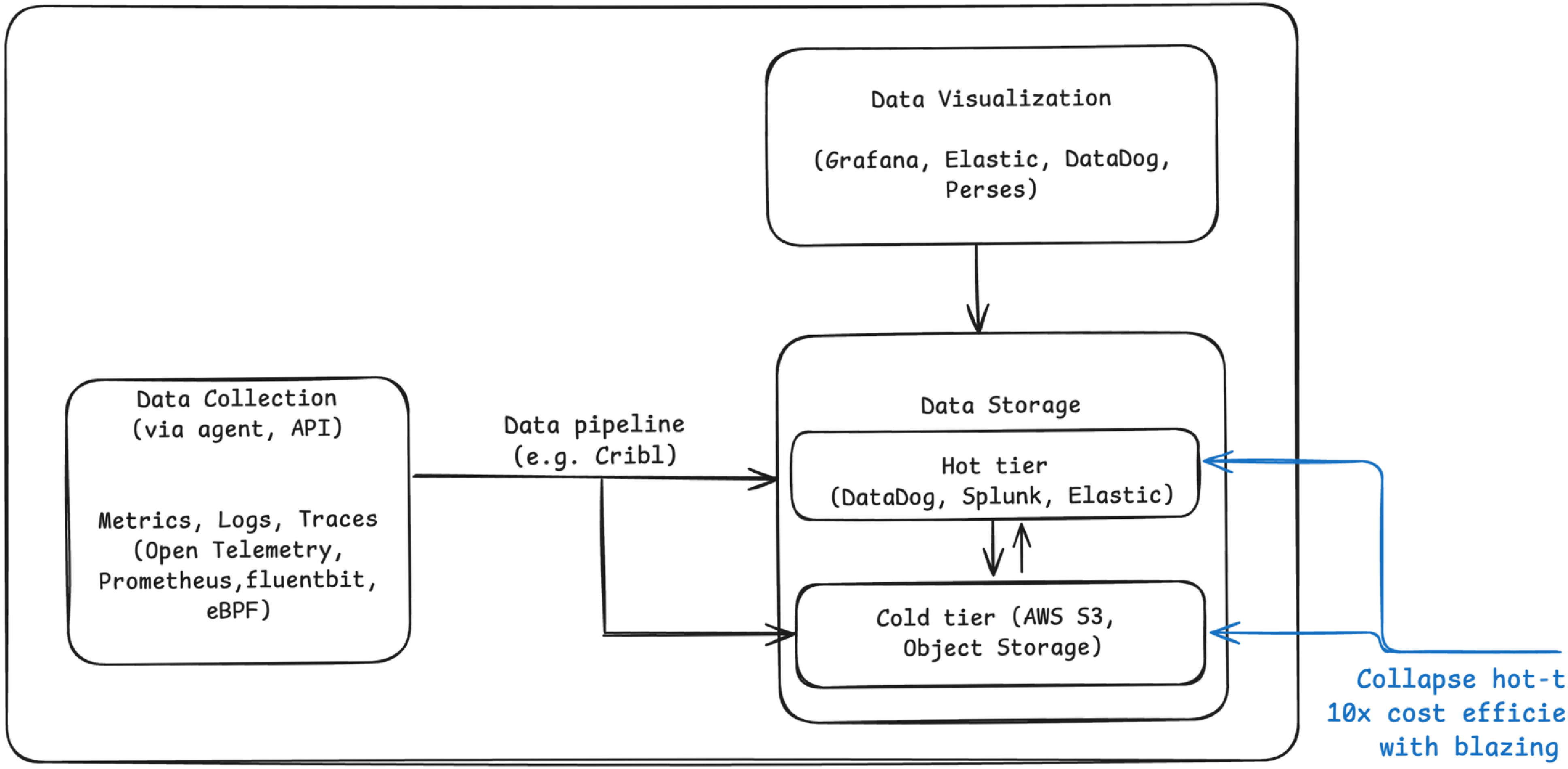
Snowflake of observability

Build a new time series database exploiting these properties

A new set of problems require a fundamentally new cloud native architecture



Everything is in S3. It's serverless. No disks at all.


Data Visualization
(Grafana, Elastic, DataDog, Perses)
Data Collection
(via agent, API)
Metrics, Logs, Traces
(Open Telemetry, Prometheus,fluentbit, eBPF)
Data pipeline
(e.g. Cribl)
Data Storage
Hot tier
(DataDog, Splunk, Elastic)
Cold tier (AWS S3, Object Storage)

Collapse hot-tier and cold-tier
10x cost efficiency and scalability
with blazing fast performance

What are the results?

Dashboard-less debugging: 5x faster

10x lower compute costs

1 Compute

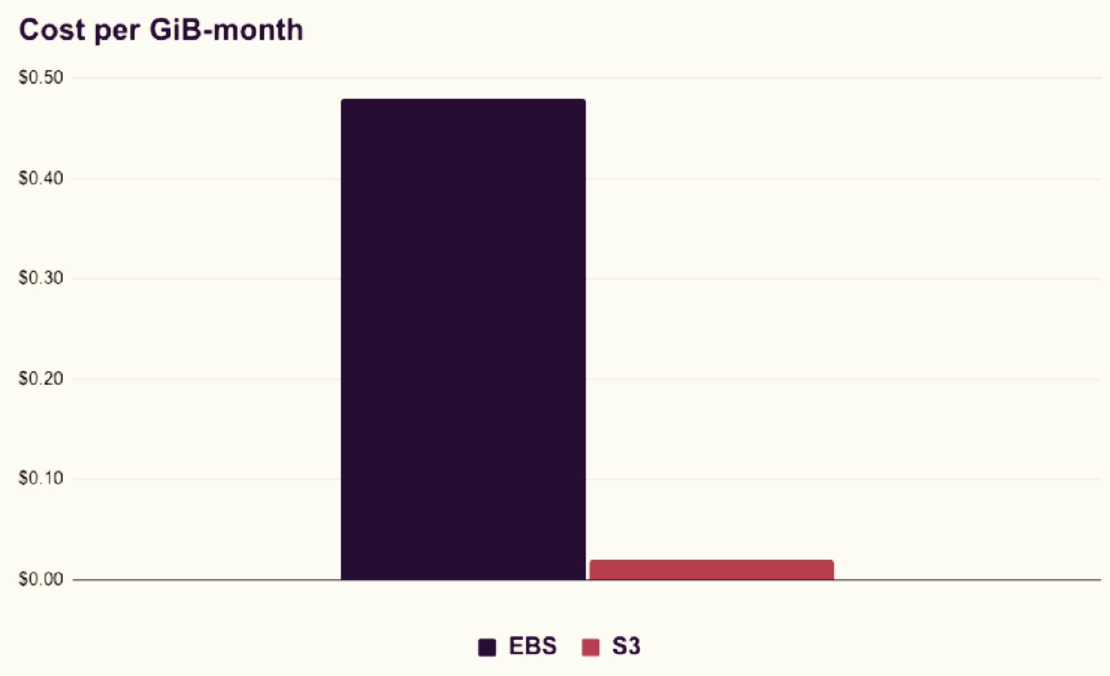
1. Serverless: don't pay for compute when not using
2. Intelligent caching => fast performance

2 Memory

1. No global indexing
2. Custom in-memory representation of metrics

24x lower storage costs

Cost per GiB-month for a 3x replicated storage system



Reference: Cloud Disks are really expensive by WarpStream

zero networking costs

1 | No replication costs

2 | Zero egress costs during ingestion with direct link

3 | Zero cross-AZ networking costs

What organizations need to do stay ahead?

1 | **Reevaluate existing observability investments**

Are current tools providing **fast, actionable insights**

Can they scale cost-effectively as systems become more complex?

2 | **Leverage AI for faster alert troubleshooting**

3 | **Prioritize cost-efficiency and scalability**

Move away from **monolithic, high-cost storage models.**

Adopt **modern cloud-native architectures**

4 | **Demand Simplicity**

Observability should be an **enabler, not an additional burden.**

1-click deployment and migration

What I spoke about today

1 | Key challenges with observability today: Cost + debugging speed

2 | Why legacy architectures won’t work for cloud native and AI workloads

3 | A new custom database built from ground up leveraging Serverless + S3 architecture

5x faster and 10x cost-efficient

4 | What organizations need to do to Stay ahead

https://play.oodle.ai

Live demo: (no login required)

Thank you!

Kiran Gollu - kiran@oodle.ai

Questions?