
aws
SUMMIT
CHICAGO | AUGUST 25, 2022

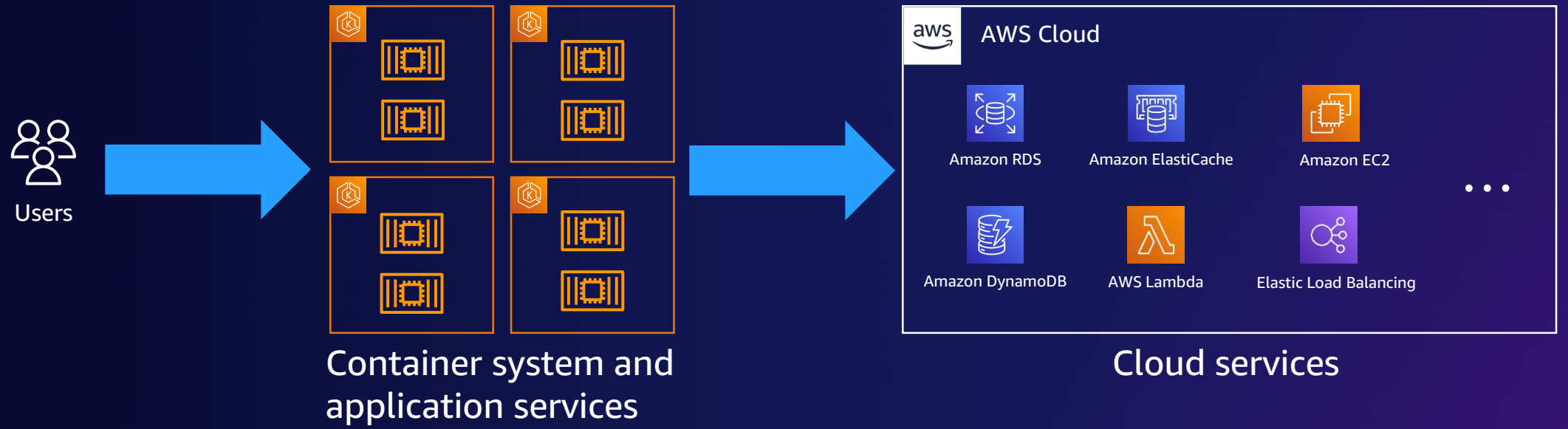
ANT301

Observing logs and traces with Amazon OpenSearch Service

Jon Handler
Sr. Principal Specialist Solutions Architect
AWS

Muthu Pitchaimani
Sr. Specialist Solutions Architect
AWS

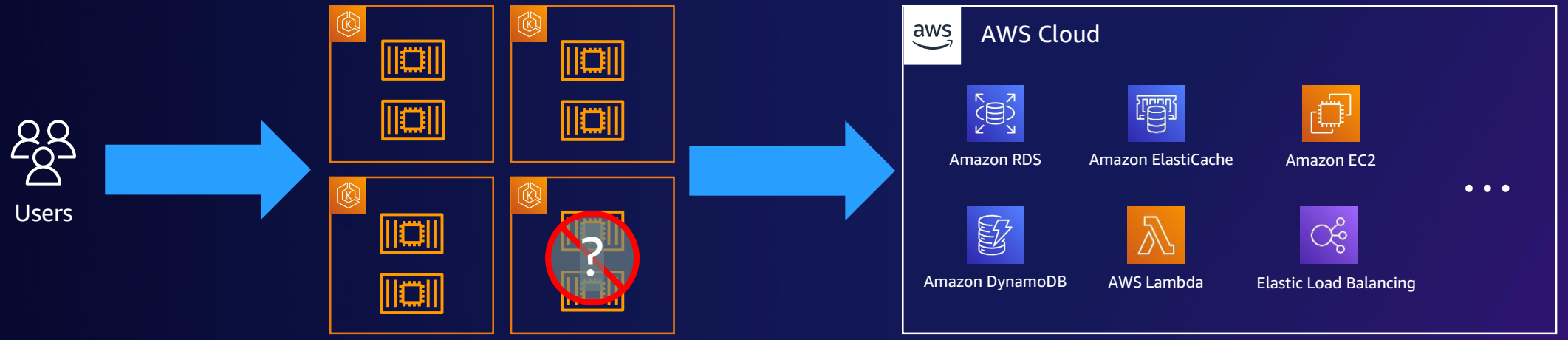
Modern applications



Distributed across services and microservices

Visibility can be low, especially for application microservice interaction and interaction with AWS services

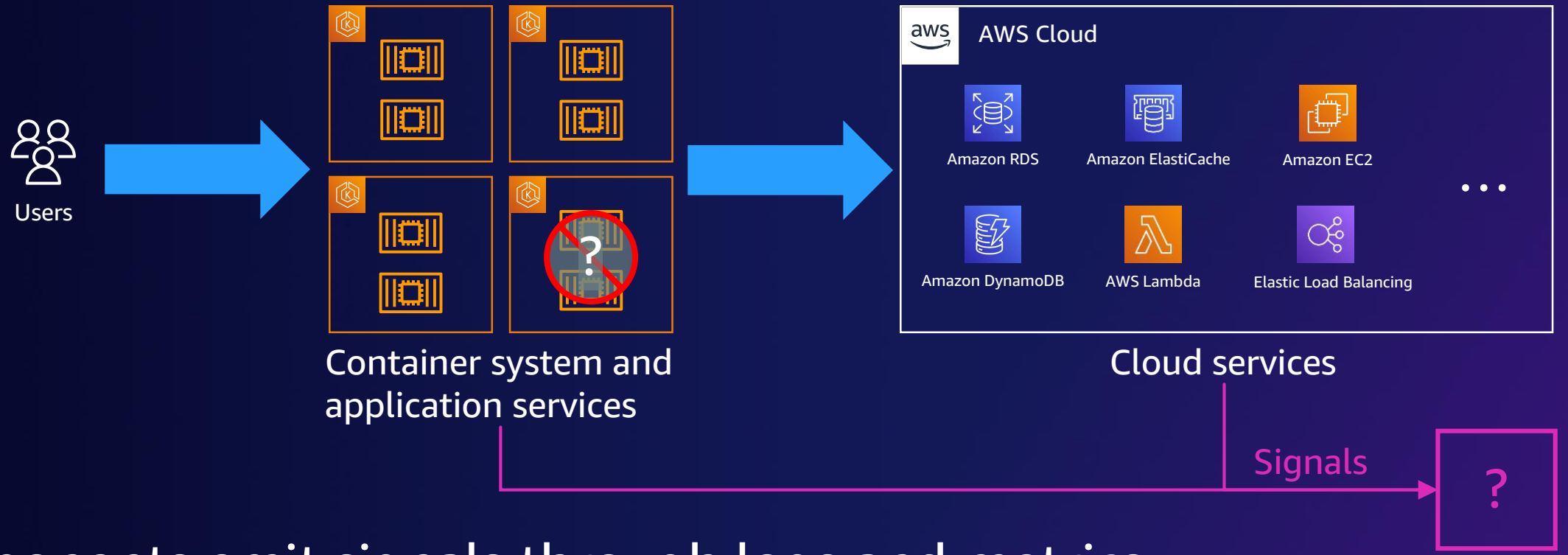
When things go wrong



Was it: a code bug? A service-API failure? A cloud service failure?

Decoupled code and services are hard to diagnose!

Gathering and analyzing signals



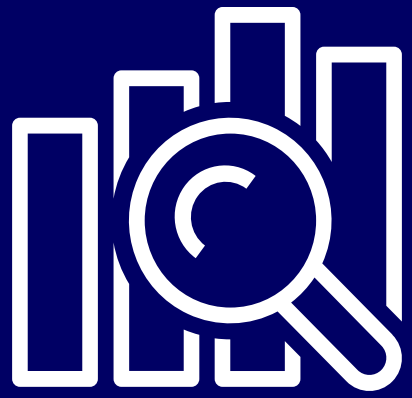
Components emit signals through logs and metrics

Remediate failures with analysis of interactions and code

OpenSearch

Community-driven, open-source **search and analytics suite** derived from Apache 2.0 licensed Elasticsearch 7.10.2 and Kibana 7.10.2

Consists of a search engine, OpenSearch; a visualization user interface, OpenSearch Dashboards; and a series functionality adding tools and plugins

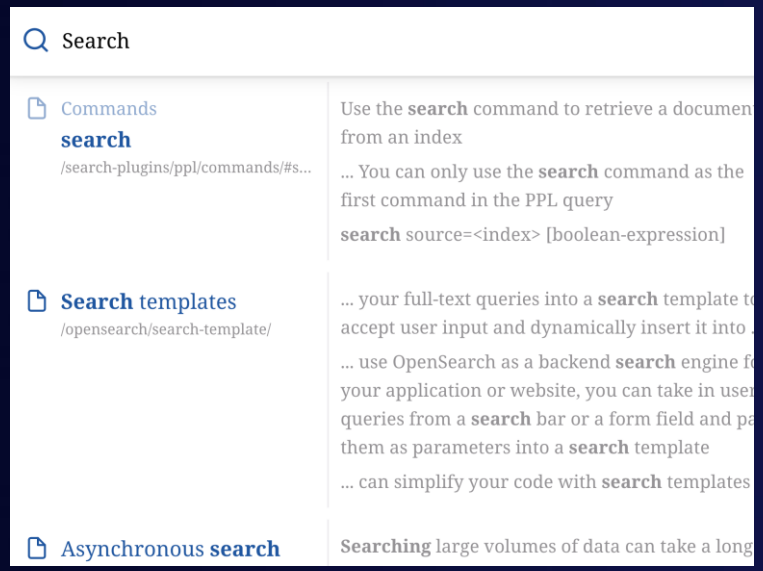


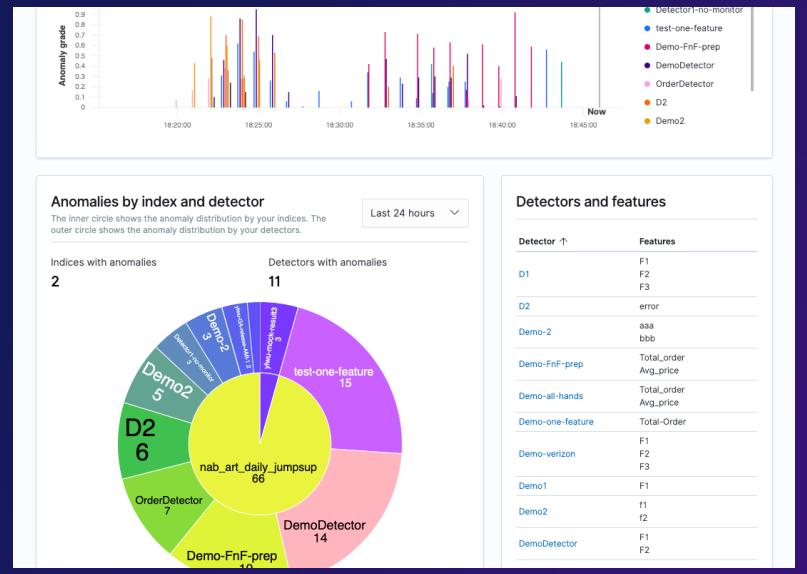
Amazon OpenSearch Service

A managed service that makes it easy to **deploy, operate,** and **scale OpenSearch** and legacy Elasticsearch clusters in the AWS Cloud

Perform interactive log analytics, real-time application monitoring, website search, and more. Offers the latest versions of OpenSearch, support for 19 versions of Elasticsearch (1.5 to 7.10 versions), and visualization capabilities powered by OpenSearch Dashboards and Kibana (1.5 to 7.10 versions).

OpenSearch





Text search

Natural language

Boolean queries

Relevance

Streaming data

High-volume ingest

Near real time

Distributed storage

Analysis

Time-based visualizations

Nestable statistics

Time series tools

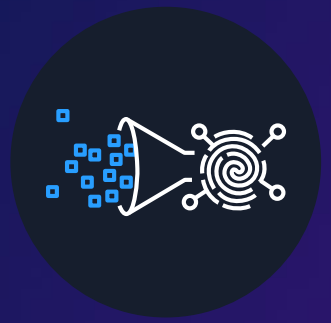
Foundation for observability: Data drives decisions

Logs

Amazon OpenSearch Service

Metrics

Amazon Managed Service for Prometheus
Amazon Managed Grafana



Traces

AWS Distro for OpenTelemetry
Fluent Bit

AWS monitoring and observability services help you maintain SLAs by **detecting, investigating,** and **remediating problems** to achieve

Availability Reliability Performance

Analyzing logs and metrics

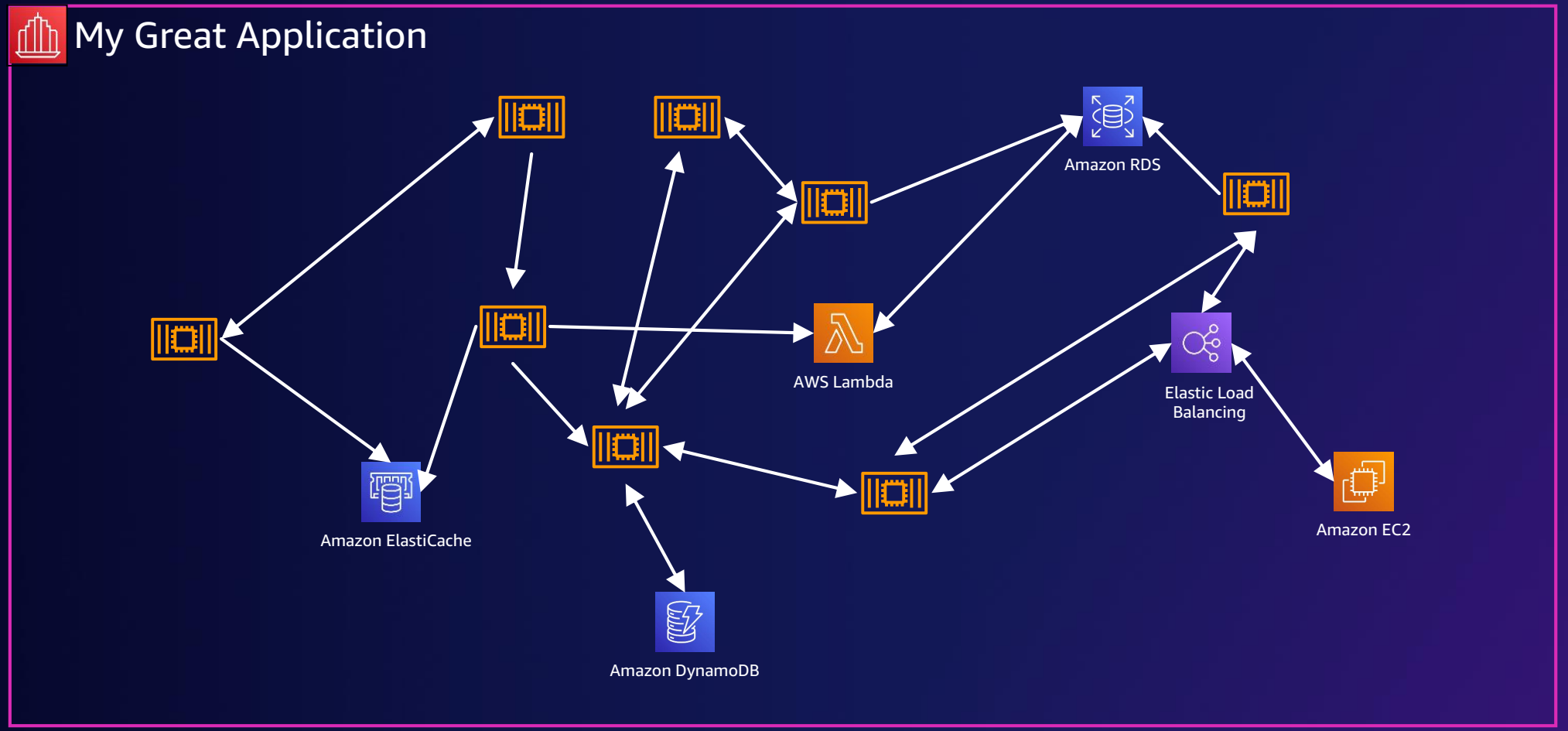
```
199.72.81.55 - - [01/Jul/1995:00:00:01 -0400] "GET /history/apollo/ HTTP/1.0" 200 6245
unicomp6.unicomp.net - - [01/Jul/1995:00:00:06 -0400] "GET /shuttle/countdown/ HTTP/1.0" 200 3985
199.120.110.21 - - [01/Jul/1995:00:00:09 -0400] "GET /shuttle/missions/sts-73/mission-sts-73.html HTTP/1.0" 200 4085
burger.letters.com - - [01/Jul/1995:00:00:11 -0400] "GET /shuttle/countdown/liftoff.html HTTP/1.0" 304 0
199.120.110.21 - - [01/Jul/1995:00:00:11 -0400] "GET /shuttle/missions/sts-73/sts-73-patch-small.gif HTTP/1.0" 200 41
burger.letters.com - - [01/Jul/1995:00:00:12 -0400] "GET /images/NASA-logosmall.gif HTTP/1.0" 304 0
burger.letters.com - - [01/Jul/1995:00:00:12 -0400] "GET /shuttle/countdown/video/livevideo.gif HTTP/1.0" 200 0
205.212.115.106 - - [01/Jul/1995:00:00:12 -0400] "GET /shuttle/countdown/countdown.html HTTP/1.0" 200 3985
d104.aa.net - - [01/Jul/1995:00:00:13 -0400] "GET /shuttle/countdown/ HTTP/1.0" 200 3985
129.94.144.152 - - [01/Jul/1995:00:00:13 -0400] "GET / HTTP/1.0" 200 7074
unicomp6.unicomp.net - - [01/Jul/1995:00:00:14 -0400] "GET /shuttle/countdown/count.gif HTTP/1.0" 200 40310
unicomp6.unicomp.net - - [01/Jul/1995:00:00:14 -0400] "GET /images/NASA-logosmall.gif HTTP/1.0" 200 786
unicomp6.unicomp.net - - [01/Jul/1995:00:00:14 -0400] "GET /images/KSC-logosmall.gif HTTP/1.0" 200 1204
d104.aa.net - - [01/Jul/1995:00:00:15 -0400] "GET /shuttle/countdown/count.gif HTTP/1.0" 200 40310
d104.aa.net - - [01/Jul/1995:00:00:15 -0400] "GET /images/NASA-logosmall.gif HTTP/1.0" 200 786
d104.aa.net - - [01/Jul/1995:00:00:15 -0400] "GET /images/KSC-logosmall.gif HTTP/1.0" 200 1204
129.94.144.152 - - [01/Jul/1995:00:00:17 -0400] "GET /images/ksclogo-medium.gif HTTP/1.0" 304 0
199.120.110.21 - - [01/Jul/1995:00:00:17 -0400] "GET /images/launch-logo.gif HTTP/1.0" 200 1713
ppptky391.asahi-net.or.jp - - [01/Jul/1995:00:00:18 -0400] "GET /facts/about_ksc.html HTTP/1.0" 200 3977
net-1-141.eden.com - - [01/Jul/1995:00:00:19 -0400] "GET /shuttle/missions/sts-71/images/KSC-95EC-0916.jpg HTTP/1.0"
ppptky391.asahi-net.or.jp - - [01/Jul/1995:00:00:19 -0400] "GET /images/launchpalms-small.gif HTTP/1.0" 200 11473
205.189.154.54 - - [01/Jul/1995:00:00:24 -0400] "GET /shuttle/countdown/ HTTP/1.0" 200 3985
waters-gw.starway.net.au - - [01/Jul/1995:00:00:25 -0400] "GET /shuttle/missions/51-l/mission-51-l.html HTTP/1.0" 200
ppp-mia-30.shadow.net - - [01/Jul/1995:00:00:27 -0400] "GET / HTTP/1.0" 200 7074
205.189.154.54 - - [01/Jul/1995:00:00:29 -0400] "GET /shuttle/countdown/count.gif HTTP/1.0" 200 40310
alyssa.prodigy.com - - [01/Jul/1995:00:00:33 -0400] "GET /shuttle/missions/sts-71/sts-71-patch-small.gif HTTP/1.0" 20
ppp-mia-30.shadow.net - - [01/Jul/1995:00:00:35 -0400] "GET /images/ksclogo-medium.gif HTTP/1.0" 200 5866
dial22.lloyd.com - - [01/Jul/1995:00:00:37 -0400] "GET /shuttle/missions/sts-71/images/KSC-95EC-0613.jpg HTTP/1.0" 20
smyth-pc.moorecap.com - - [01/Jul/1995:00:00:38 -0400] "GET /history/apollo/apollo-13/images/70HC314.GIF HTTP/1.0" 20
205.189.154.54 - - [01/Jul/1995:00:00:40 -0400] "GET /images/NASA-logosmall.gif HTTP/1.0" 200 786
ix-orl2-01.ix.netcom.com - - [01/Jul/1995:00:00:41 -0400] "GET /shuttle/countdown/ HTTP/1.0" 200 3985
```

System-level view, from metrics and information in log lines

Error location through messages in logs

Missing: interaction between subsystems

What about service interaction?



What a mess!

OpenTelemetry

- OpenTelemetry is a community-driven, open-source project designed for the standardization, creation, and management of telemetry data such as traces, metrics, and logs
- Supports many popular open-source wire formats including Jaeger, Zipkin, and Prometheus
- Currently supports traces (GA), metrics (preview), and logs (experimental)
- Traces are emitted as a log line, either intra- or inter-service

AWS Distro for OpenTelemetry

OpenTelemetry

A Cloud Native Computing Foundation project

Open source observability agents and libraries

Supports all 3 data signals in 11 languages

AWS Distro for OpenTelemetry

Secure, production-ready, open-source distribution supported by AWS

Code contributions are upstream in OpenTelemetry

Certified by AWS for security and predictability

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Data Prepper

- Data Prepper is a community-driven, open-source data collector for processing observability data
- Provides features to filter, enrich, transform, normalize, and aggregate data for downstream analytics and visualization
- Currently supports processing of distributed trace data and log ingestion with plans to support metric data in the future
- Integrations with Jeager, Zipkin, OpenTelemetry, and Fluent Bit

Traces and spans

A span is a unit of work

Spans have

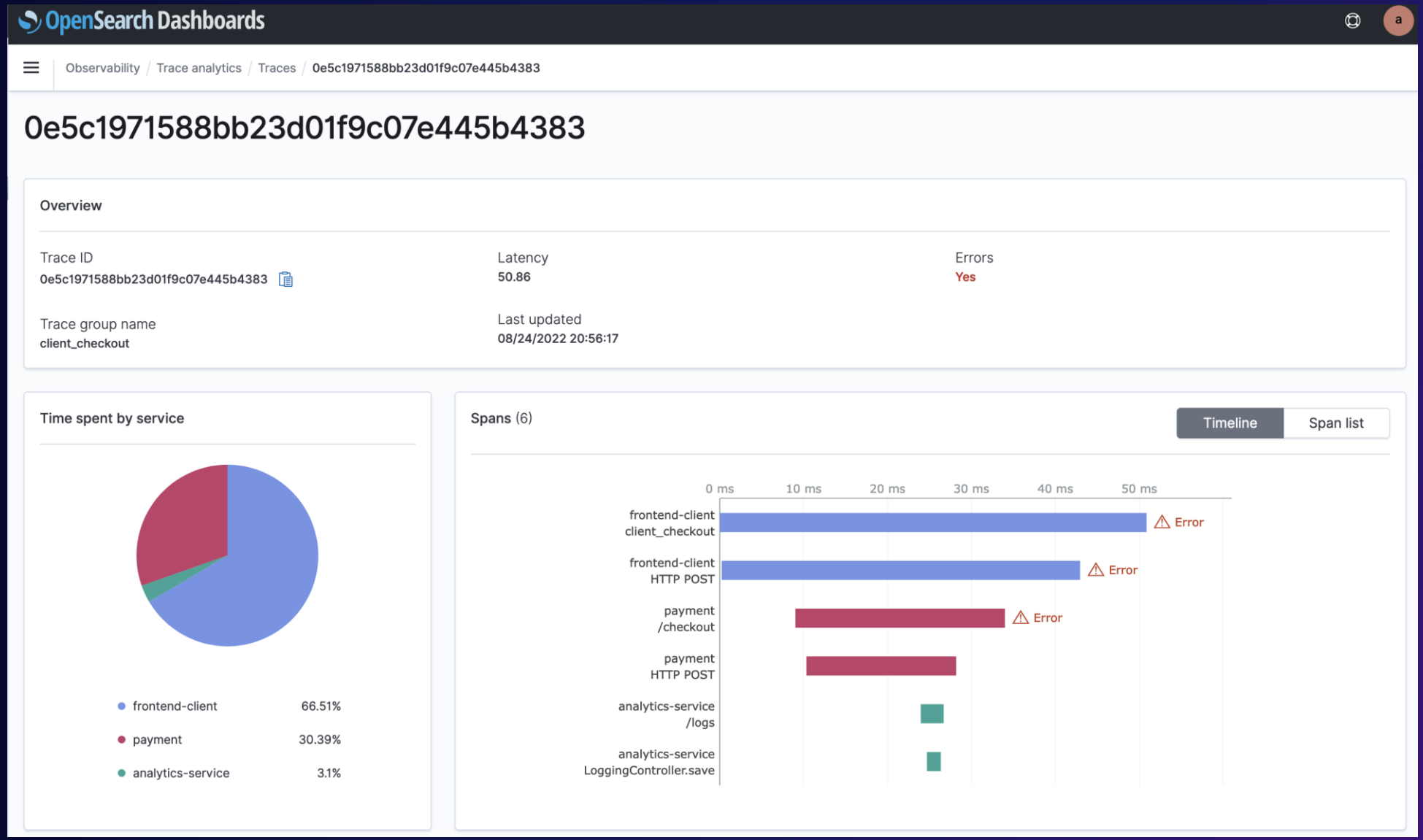
- Duration
- Methods
- Status codes

A trace is a particular user request

Traces have spans


```
{
  "_index" : "otel-v1-apm-span-000001",
  "_type" : "_doc",
  "_id" : "520f74be0ad94c8c",
  "_score" : 1.0,
  "_source" : {
    "traceId" : "348c4305a3418a1ea5631de518bb5cf7",
    "spanId" : "520f74be0ad94c8c",
    "traceState" : "",
    "parentSpanId" : "89e79a790afc003a",
    "name" : "HTTP PUT",
    "kind" : "SPAN_KIND_CLIENT",
    "startTime" : "2022-08-23T01:33:47.482462751Z",
    "endTime" : "2022-08-23T01:33:47.532808718Z",
    "durationInNanos" : 50345967,
    "serviceName" : "order",
    "events" : [ ],
    "links" : [ ],
    "droppedAttributesCount" : 0,
    "droppedEventsCount" : 0,
    "droppedLinksCount" : 0,
    "traceGroup" : "client_cancel_order",
    "traceGroupFields.endTime" : "2022-08-23T01:33:47.578544903Z",
    "traceGroupFields.statusCode" : 0,
    "traceGroupFields.durationInNanos" : 107274133,
    "span.attributes.http@method" : "PUT",
    "span.attributes.http@url" : "http://database-service.database-service.svc.cluster.local:80/cart_empty",
    "resource.attributes.telemetry@sdk@name" : "opentelemetry",
    "instrumentationLibrary.version" : "0.28b1",
    "resource.attributes.telemetry@sdk@language" : "python",
    "resource.attributes.telemetry@sdk@version" : "1.9.1",
    "resource.attributes.service@instance@id" : "139947915721696",
    "resource.attributes.service@name" : "order",
    "resource.attributes.host@hostname" : "order-service-6bc467f446-6kmtd",
    "span.attributes.http@status_code" : 200,
    "status.code" : 0,
    "instrumentationLibrary.name" : "opentelemetry.instrumentation.requests"
  }
},
```

A single trace contains multiple spans



A single span can have MULTIPLE children

A single span can be of different kinds

- SERVER
  - Describes server-side handling of a synchronous remote request
  - Is the child of a remote CLIENT span
- CLIENT
  - Describes a request to some remote service
  - Is the parent of a remote SERVER span and does not end until the response is received
- PRODUCER
  - Describes initiators of an asynchronous request
  - Will often end before the corresponding child CONSUMER span
- CONSUMER
  - Describes a child of an asynchronous PRODUCER request
- INTERNAL (Default)
  - Describes internal operation within an process

Amazon OpenSearch Service

The what and why of search

If you have an **e-commerce platform,** you want customers to find the product they are looking for quickly

If you have a **document portal** with documents like scientific research articles, investment analyses, or health records, you want to enable a speedy and relevant search experience for your users

You may want to increase user engagement on your platform by **delivering personalized recommendations,** like a weekly music playlist or food recipes

Beyond these examples, you may have other parts of your tech stack where you want to add an **easy-to-use and snappy search experience,** especially with the option to integrate machine learning capabilities to power a personalized experience

Logs are ubiquitous

Applications and infrastructure

Services/microservices

Web applications

Business applications

APIs

IT and DevOps

Databases

Load balancers

Networking

Servers

IoT and wireless

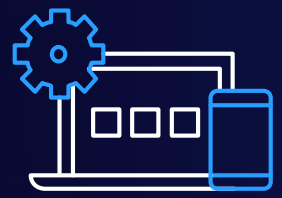
Automotive

Home devices

Manufacturing

Mobile applications

Turning logs into insights



Applications

Is my infrastructure working?

What is the latency and error rate?

What caused my application issue?

Security

Is there any suspicious authentication activity?

What data was accessed by this IP address?

Are there instances of fraud?

Business insights

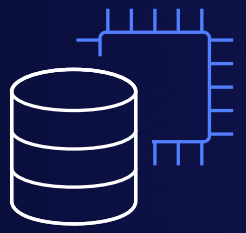
What content/products are my users interested in?

Which features are used most or least?

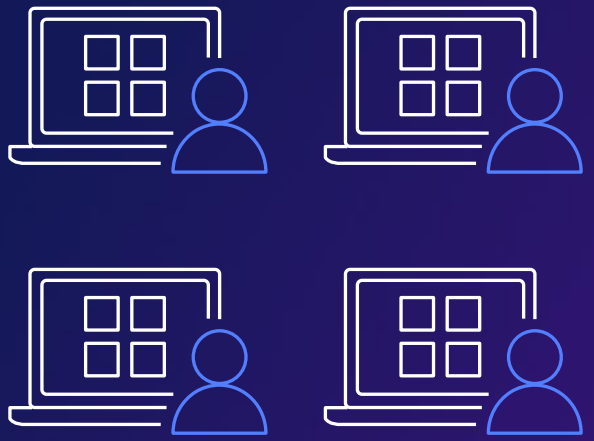
What users are most active and why?

How does it work?

Server, application, network, AWS, and other logs

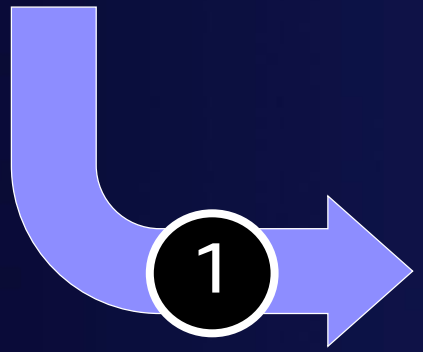


Application data



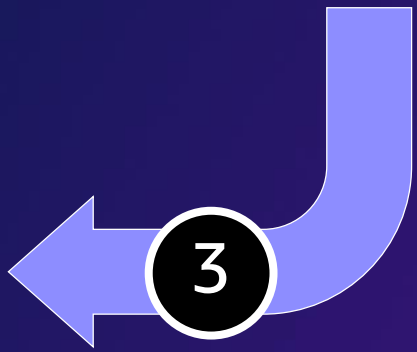
Application users, analysts, DevOps, and security

Indexed and instantly searchable



JSON documents

Amazon OpenSearch Service domain



Rest APIs, clients, and OpenSearch Dashboards

Amazon OpenSearch Service multi-layer security

Integrate with SAML and Amazon Cognito for OpenSearch Dashboards login

IAM to control access to the endpoint

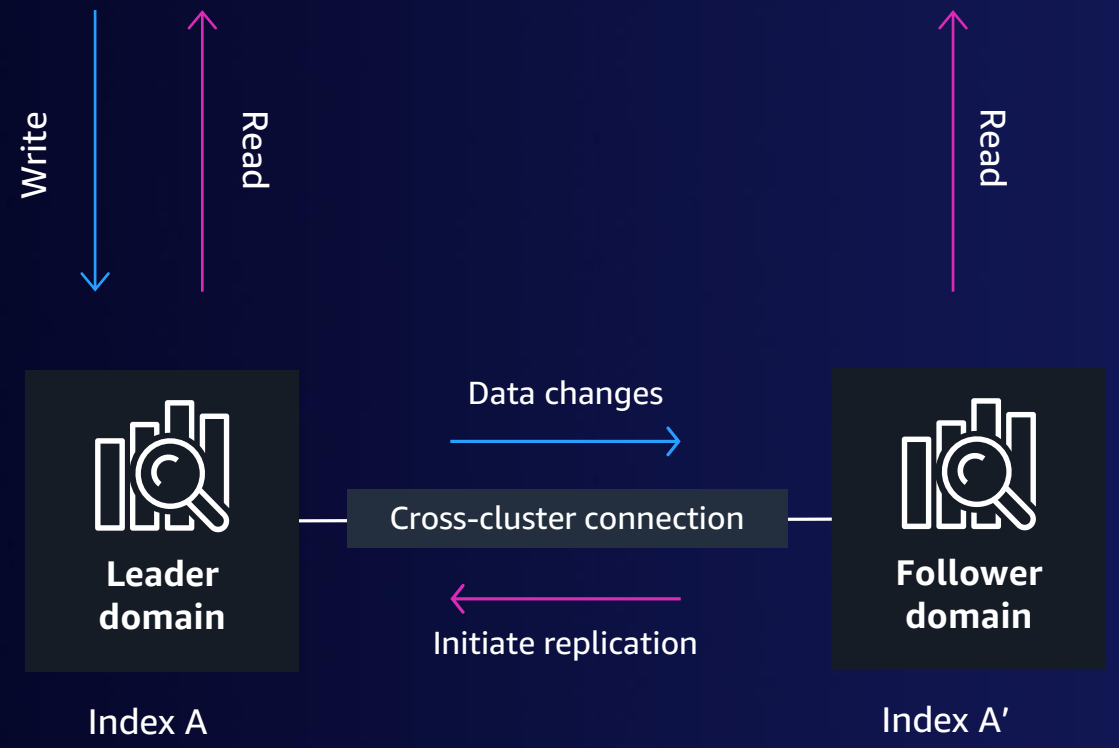
Use a private endpoint to deploy into your VPC and security groups for traffic control

Use OpenSearch fine-grained access control to secure your data and dashboards

Encrypt your data, in flight and at rest

Cross-cluster replication

HIGH AVAILABILITY, DISASTER RECOVERY, AND DATA PROXIMITY



Make your data highly available

- Active-passive replication
- Granular control at index level
- Cross-account and cross-Region support
- Sequential consistency and sub-minute global replication
- One leader/many followers

Machine learning innovations in OpenSearch

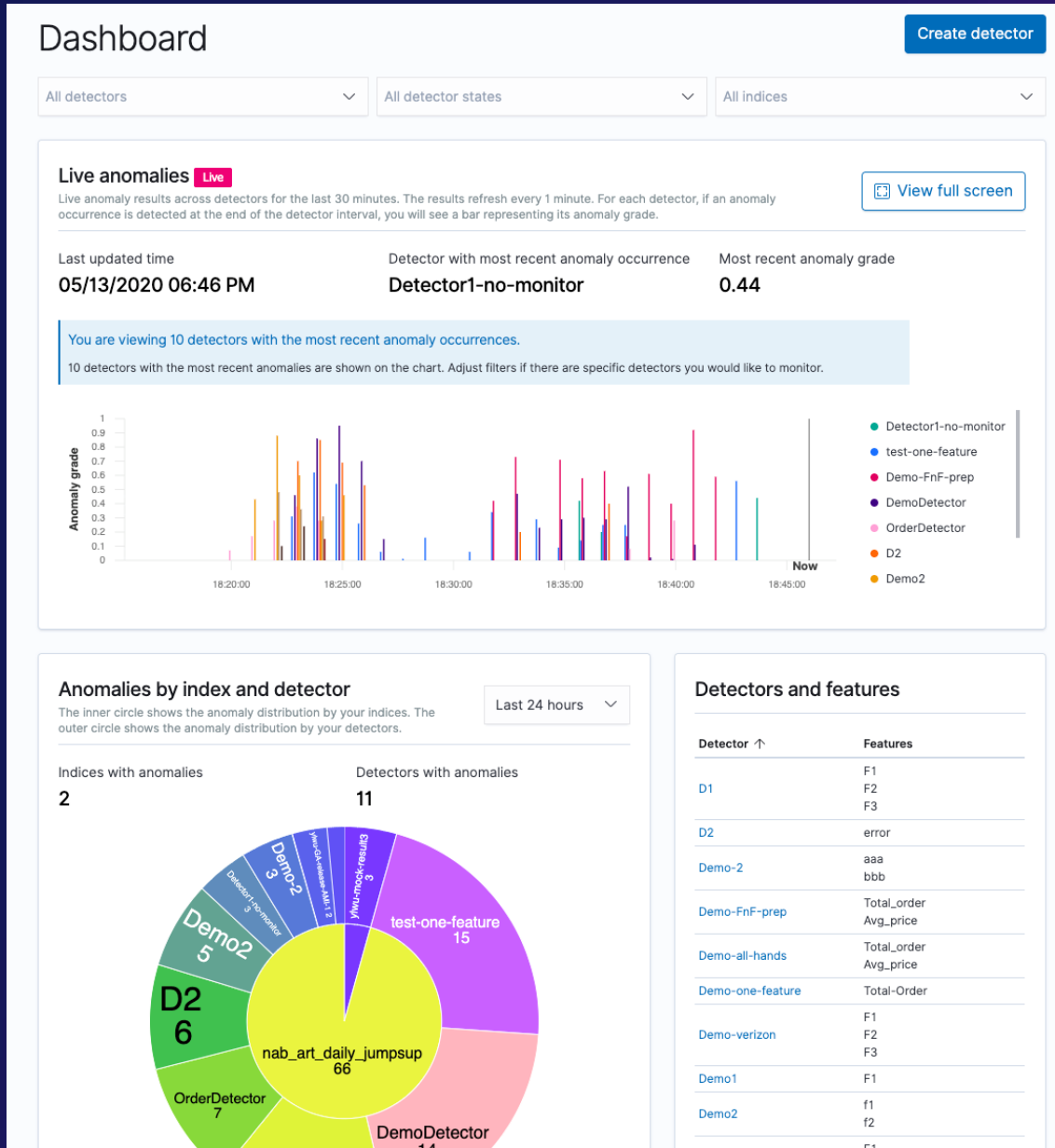
ANOMALY DETECTION FOR TIME SERIES

Mitigate issues faster with anomaly detection in streaming data

Performant at scale – machine learning models are distributed and processed across nodes

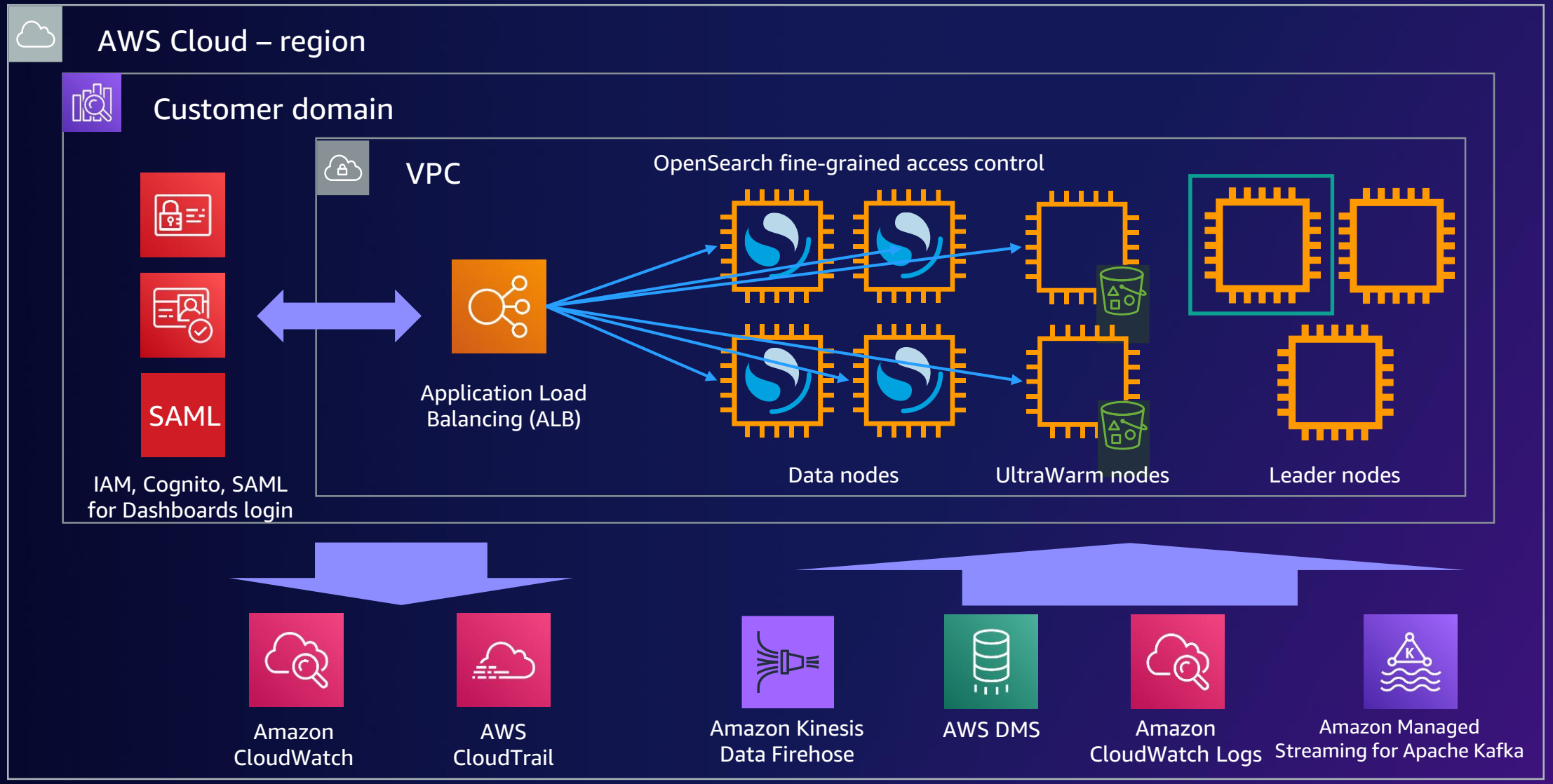
Easy to use – machine learning expertise is not required to use the service

Based on Random Cut Forest (RCF): Guha, Mishra, Roy, Schrijvers ICML 2016

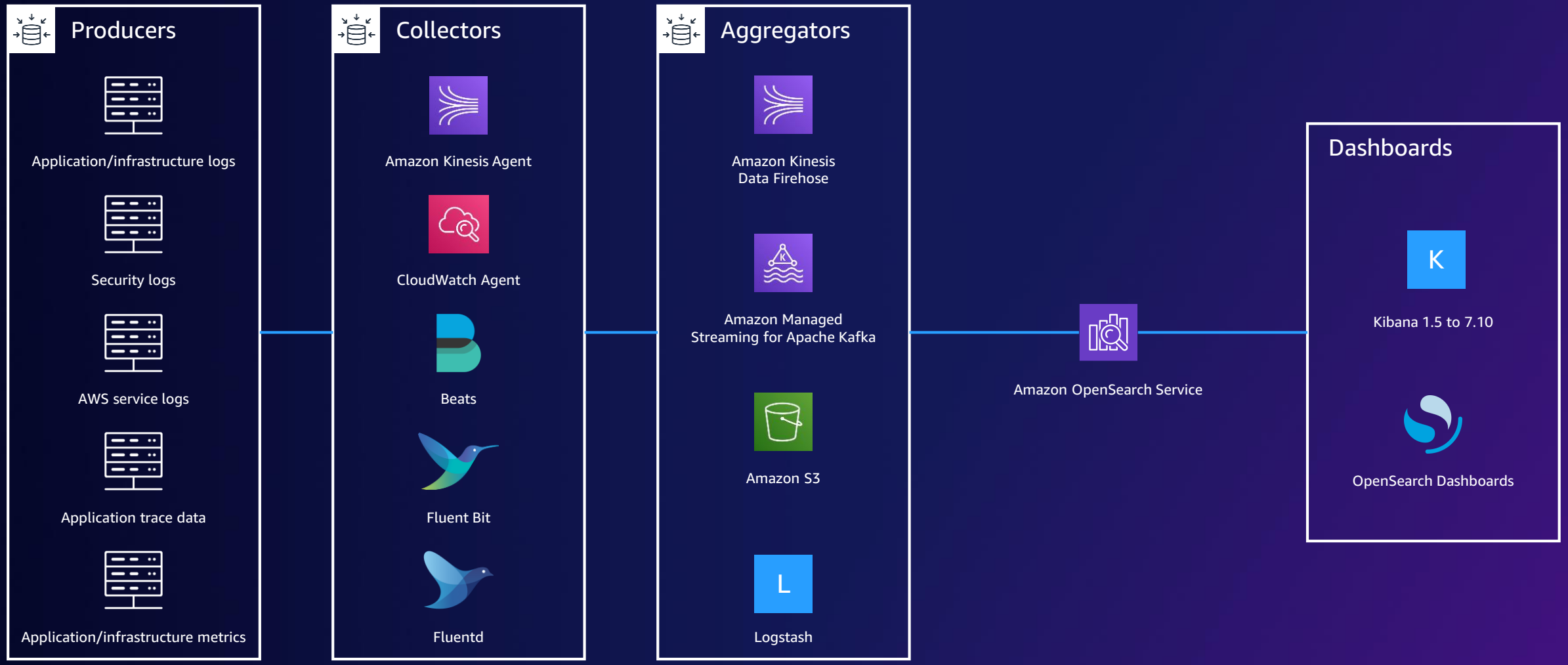


Architectures

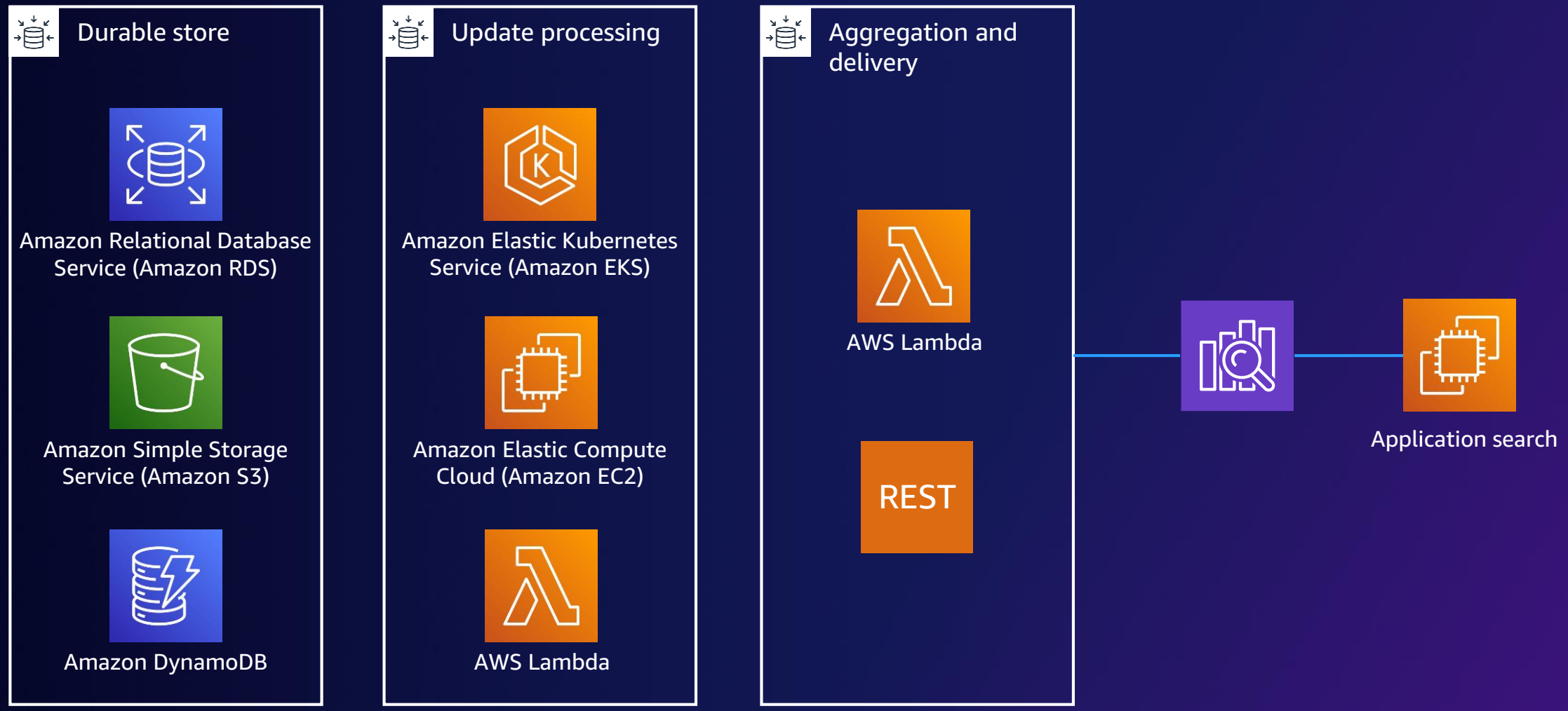
Amazon OpenSearch Service architecture



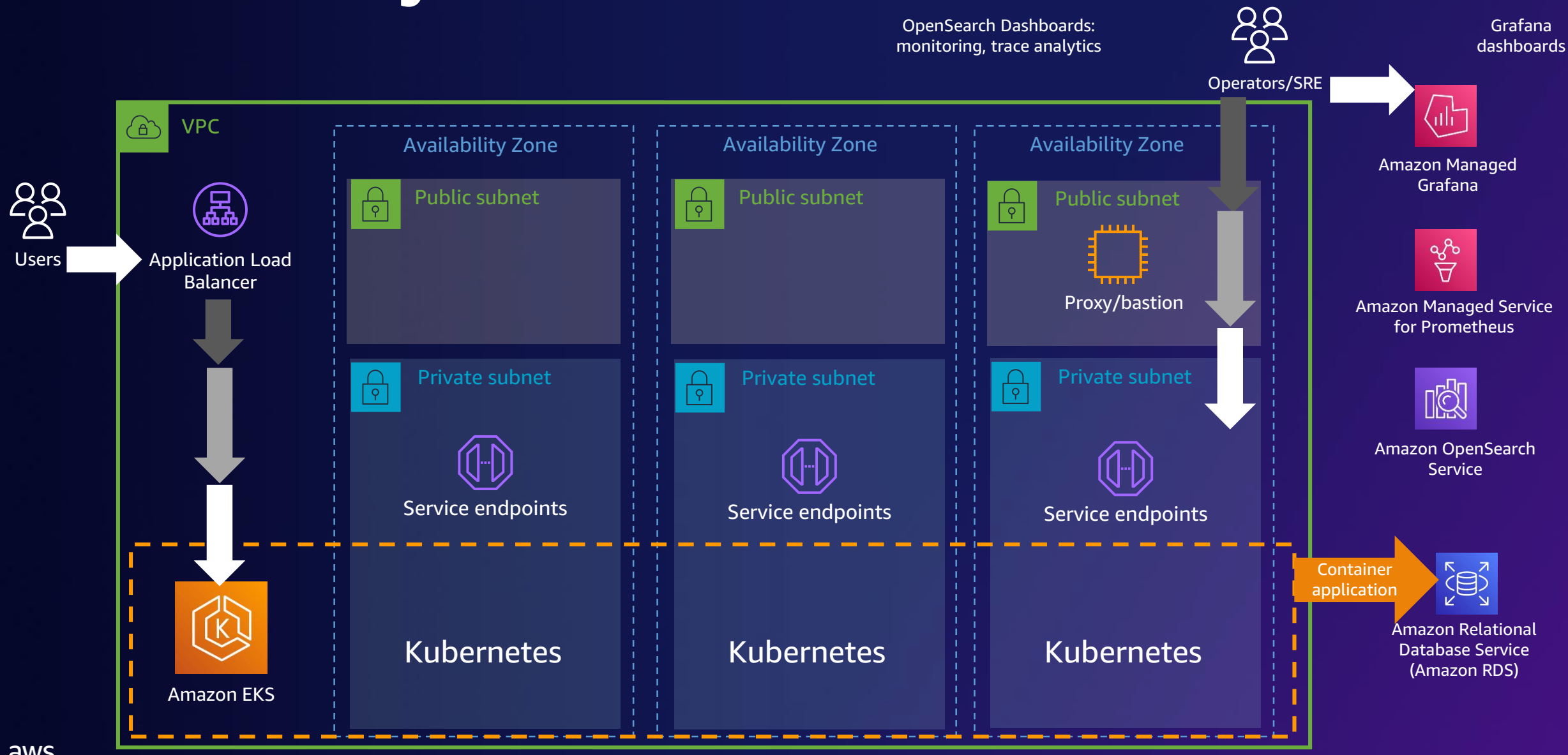
Amazon OpenSearch Service logs ingestion flow



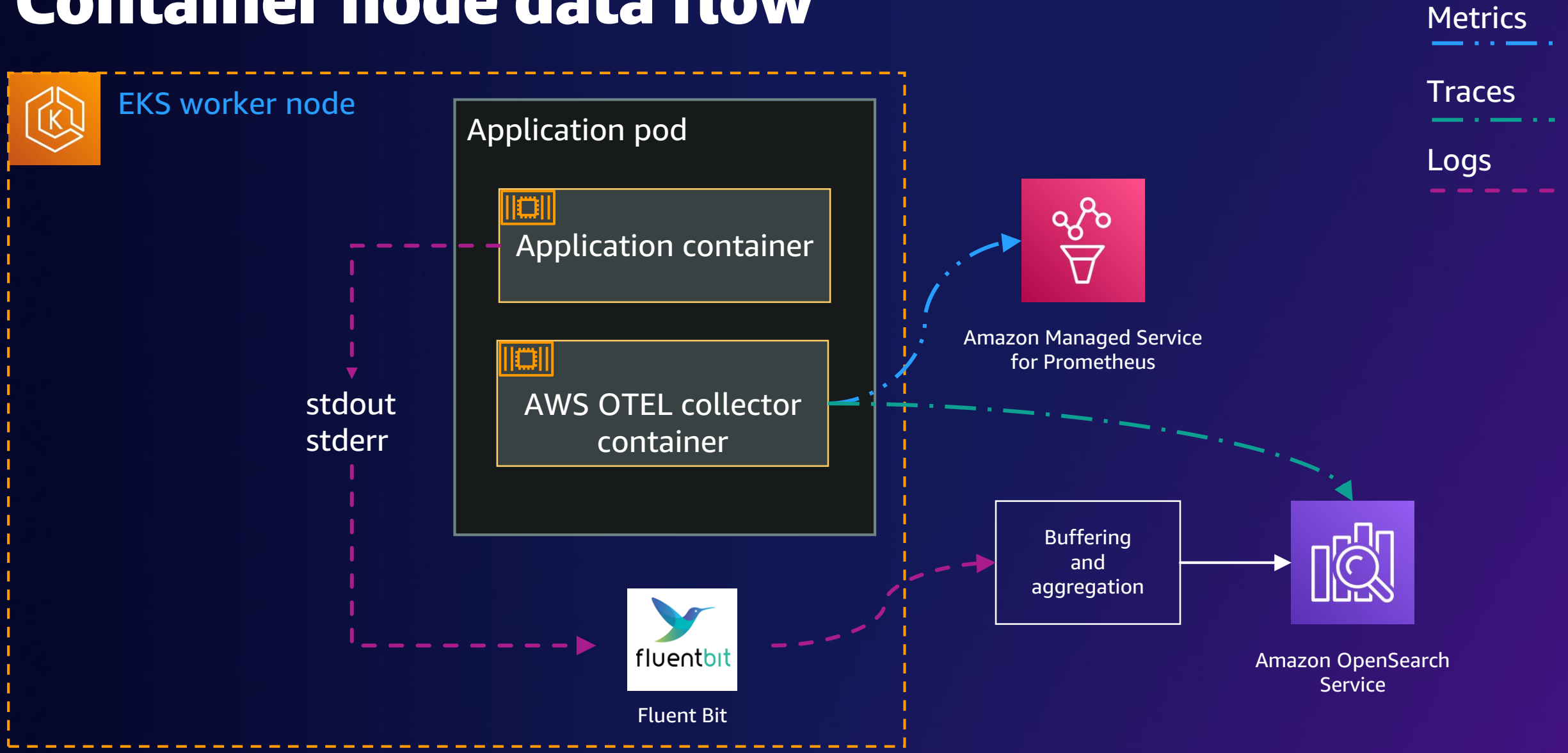
Amazon OpenSearch Service search ingestion flow



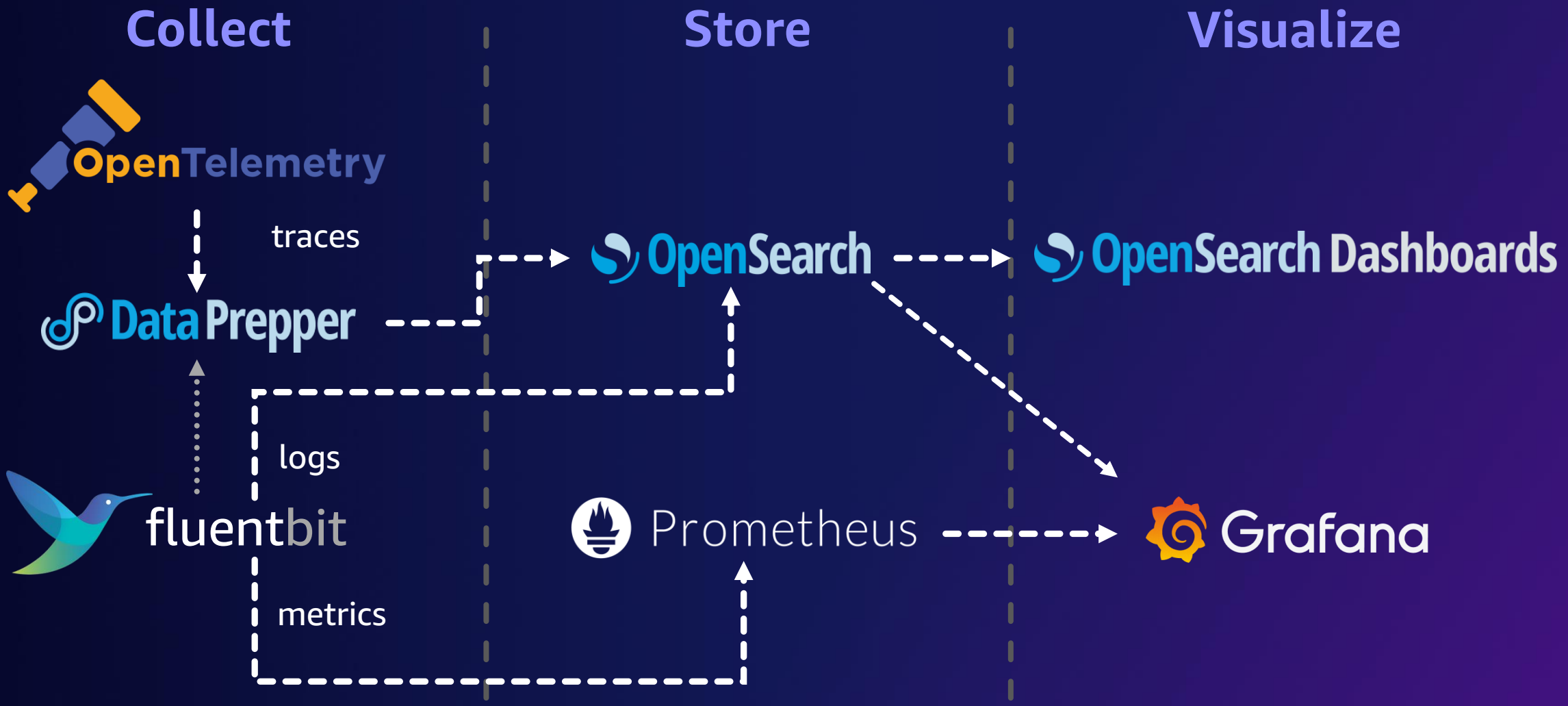
Observability architecture with AWS services



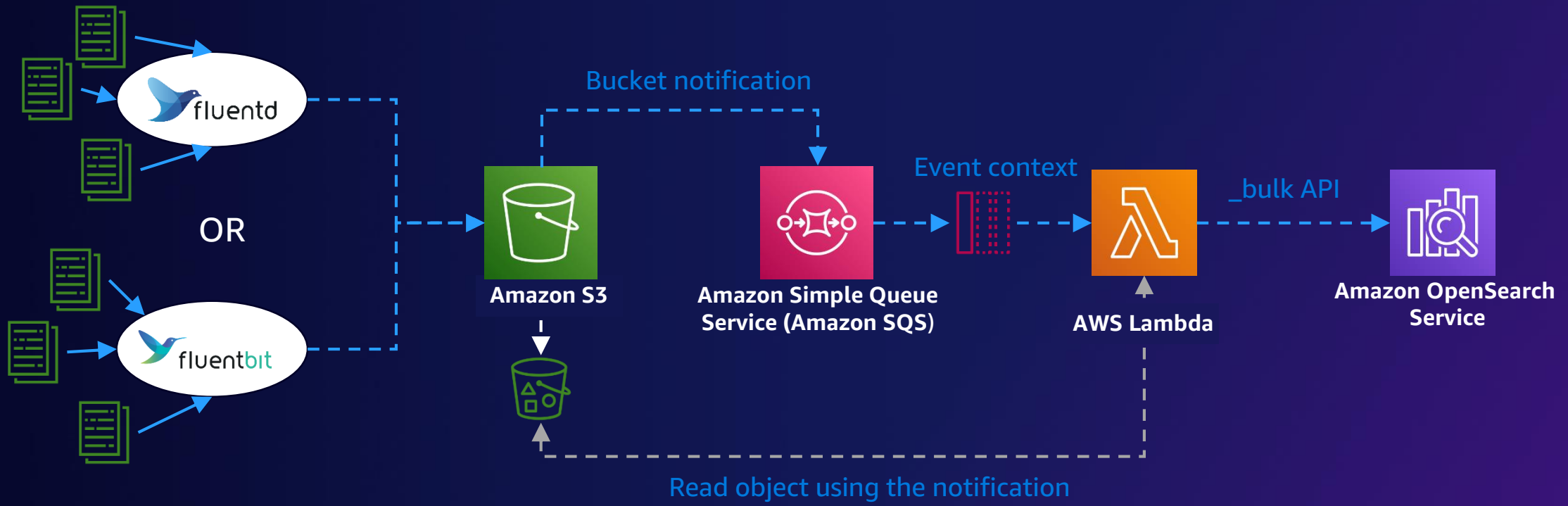
Container node data flow



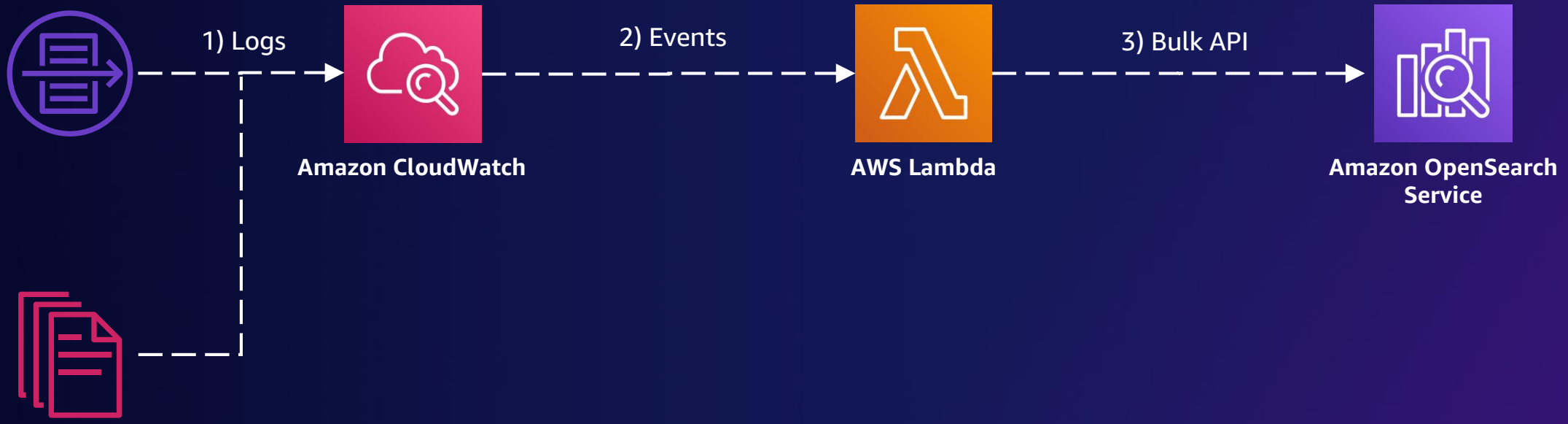
Open source observability architecture



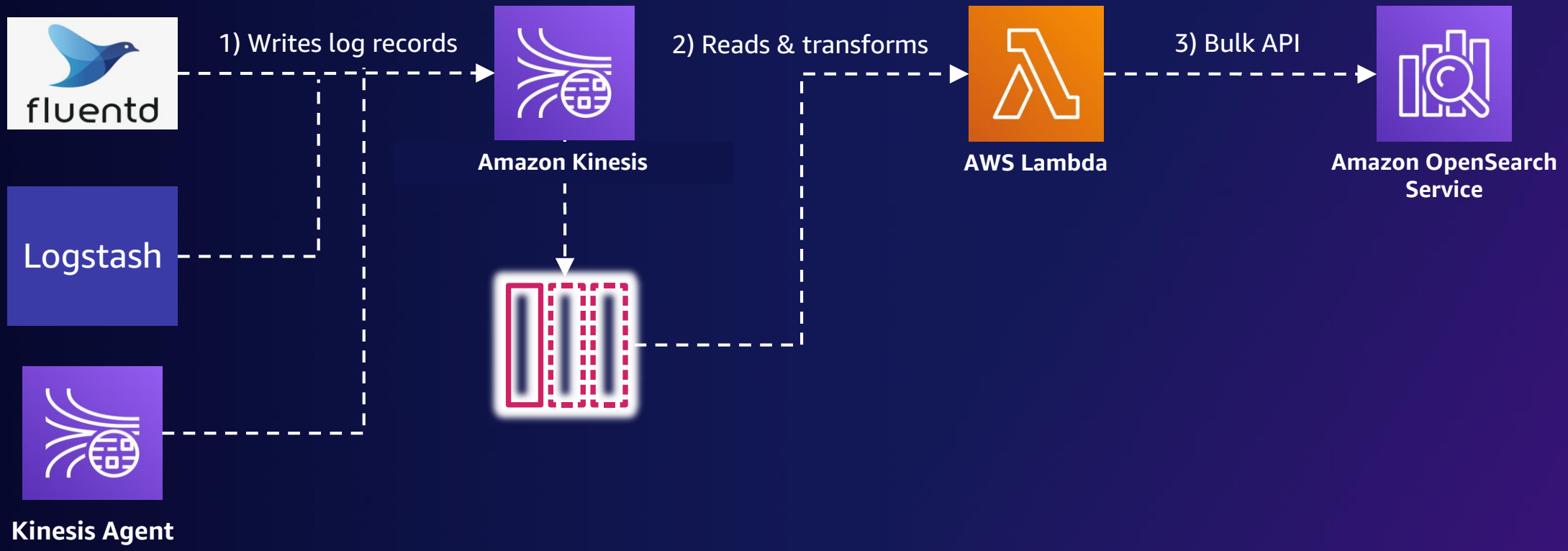
Low-cost buffering and delivery



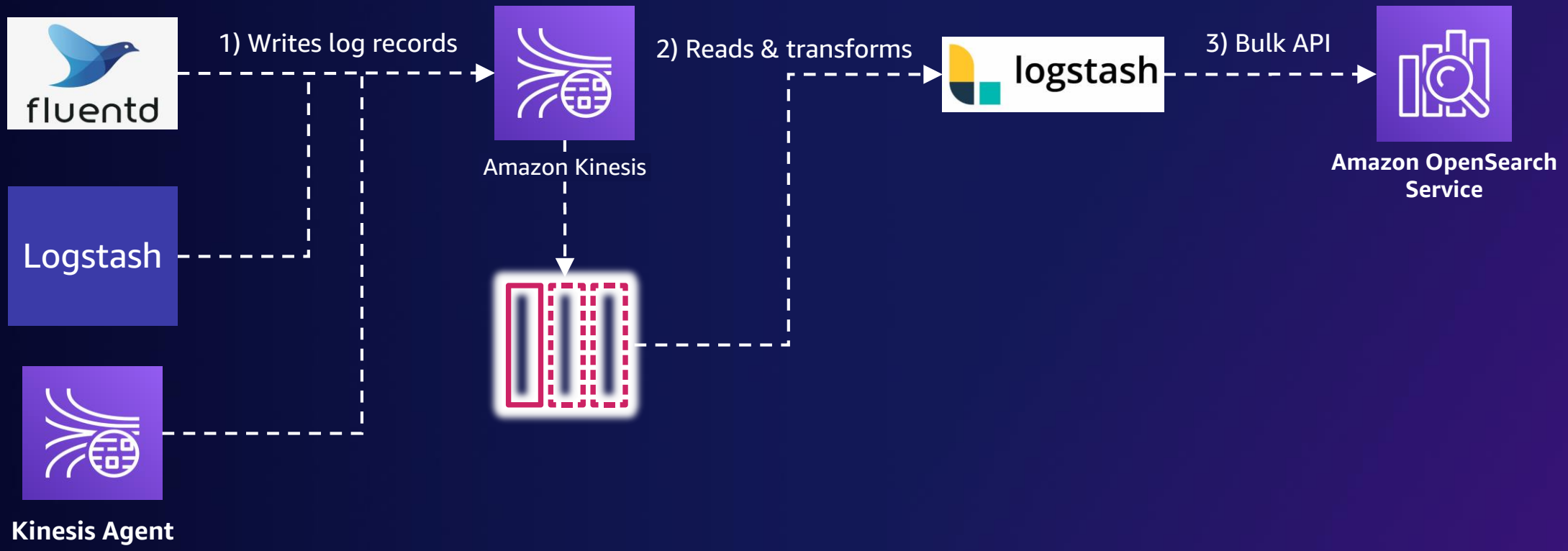
Amazon CloudWatch logs approach



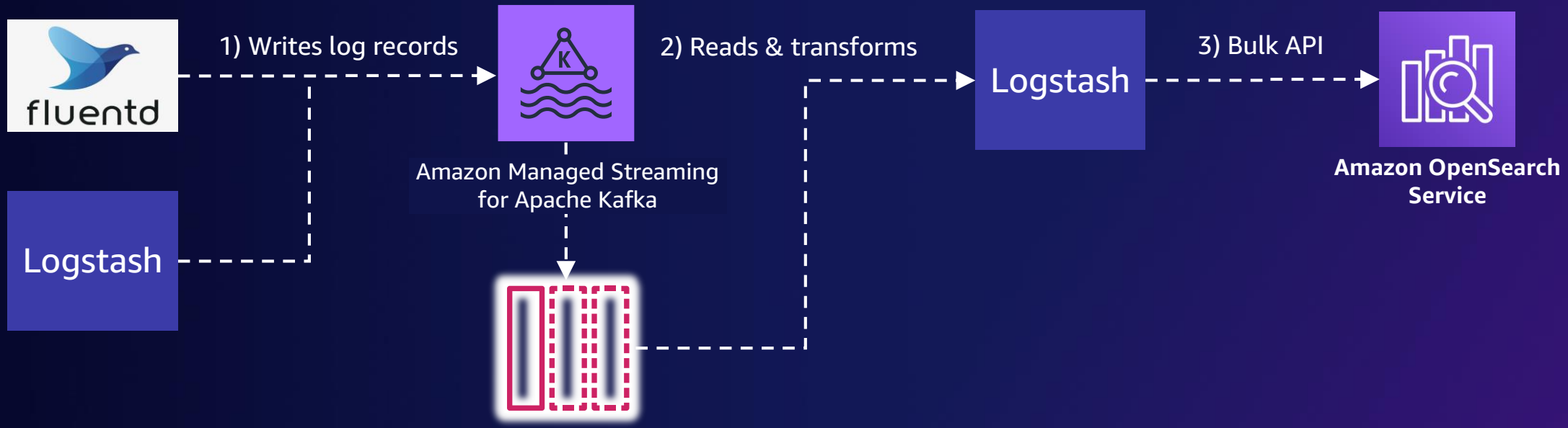
Amazon Kinesis – AWS Lambda approach



Amazon Kinesis – Logstash approach

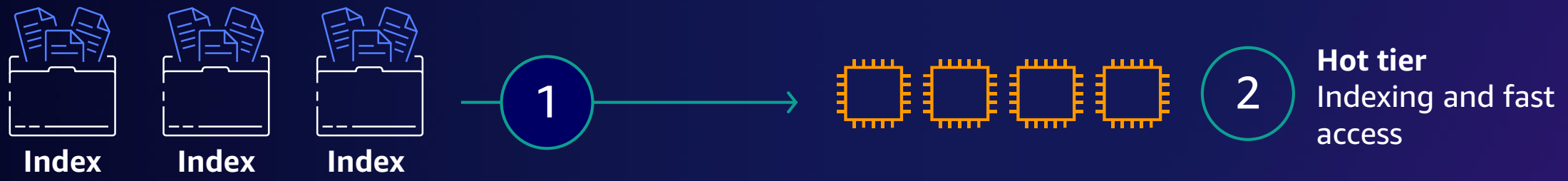


Amazon MSK – Logstash approach



Data management

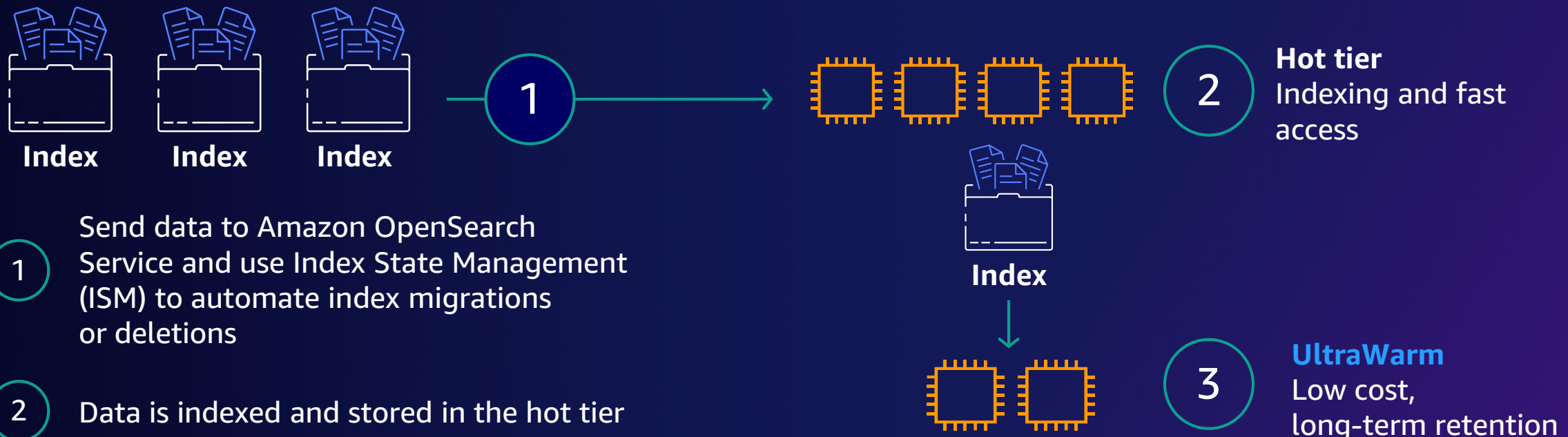
Data lifecycle in Amazon OpenSearch Service



1. Send data to Amazon OpenSearch Service and use Index State Management (ISM) to automate index migrations or deletions

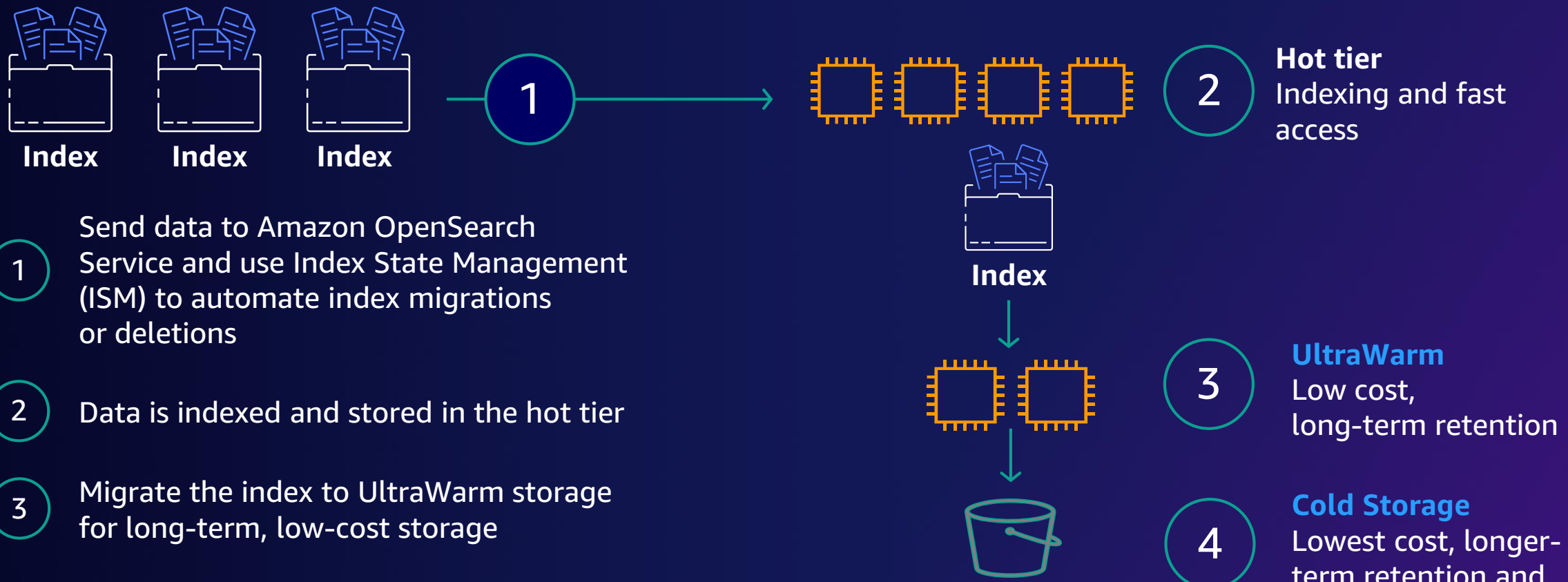
2. Data is indexed and stored in the hot tier

Data lifecycle in Amazon OpenSearch Service



1. Send data to Amazon OpenSearch Service and use Index State Management (ISM) to automate index migrations or deletions
2. Data is indexed and stored in the hot tier
3. Migrate the index to UltraWarm storage for long-term, low-cost storage

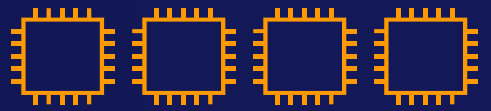
Data lifecycle in Amazon OpenSearch Service



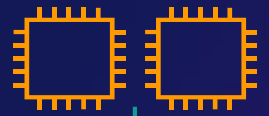
1. Send data to Amazon OpenSearch Service and use Index State Management (ISM) to automate index migrations or deletions
2. Data is indexed and stored in the hot tier
3. Migrate the index to UltraWarm storage for long-term, low-cost storage
4. Store data in cold storage for longer-term, lowest-cost storage

Data lifecycle in Amazon OpenSearch Service

1



2 **Hot tier**
Indexing and fast access



3 **UltraWarm**
Low cost, long-term retention

4 **Cold Storage**
Lowest cost, longer-term retention and on-demand access

5

1. Send data to Amazon OpenSearch Service and use Index State Management (ISM) to automate index migrations or deletions
2. Data is indexed and stored in the hot tier
3. Migrate the index to UltraWarm storage for long-term, low-cost storage
4. Store data in cold storage for longer-term, lowest-cost storage
5. Delete the index at end-of-life

aws

Sizing and capacity planning

Capacity planning

Gather information

- Data per day
- Number of indexes
- Concurrency
- Rps (Docs, Qs)
- Index/query complexity
- Durability need

Plan for storage

- Ephemeral/EBS
- Replication
- Retention
- UltraWarm
- Cold storage

Storage is a solid starting point

Plan for compute

- Concurrency – number of readers/writers
- Active indexes/shards
- Total request counts and latencies

CPUs = 1.5 * active shards

Your configuration determines capacity

Instance type

- Deploy instances based on storage and compute needs

Instance count

- Add instances for increased parallelism

Storage

- Index data (primary and replica shards) is stored on disk

Shard count

- Shards are the units of work and storage

Logs workloads are storage-driven. Search workloads are CPU/JVM-driven

Step 1: figure out storage need

Storage needed =

Source/day * 1.1 * 2 * retention * 1.15

Search: 100 GB of data needs 250 GB of storage

Logs: 1 TB daily of source data needs 18 TB of storage for 7 days of retention

Step 2: figure out shard count

Primary shards =

Index size / target shard size

Logs, use 50 GB max.
Search evaluate 20–30 GB

Step 3: Set a template

```
*PUT
<endpoint>/_template/template1
{
    "index_patterns": [”logs*"],
    "settings": {
        "number_of_shards": 50,
        "number_of_replicas": 1
    }
}
```

Step 4: Adjust for usage

vCPU = 1.5 * active shards

Active shards

Primaries for queries

Primaries and replicas for updates

E.g., 4 data streams at 1 TB daily means 40 total shards (20 primary and 20 replica) active, so make sure to have 64 total CPUs

Example

100 GB of logs per day

1 index pattern

7 days hot

23 days UltraWarm

335 days Cold

Total hot storage

100 * 1.1 * 7 * 2 * 1.15 = 1.77 TiB

Min CPUs hot = 6

Total Warm Storage

100 * 1.1 * 23 = 2.53 TiB

Total Cold Storage

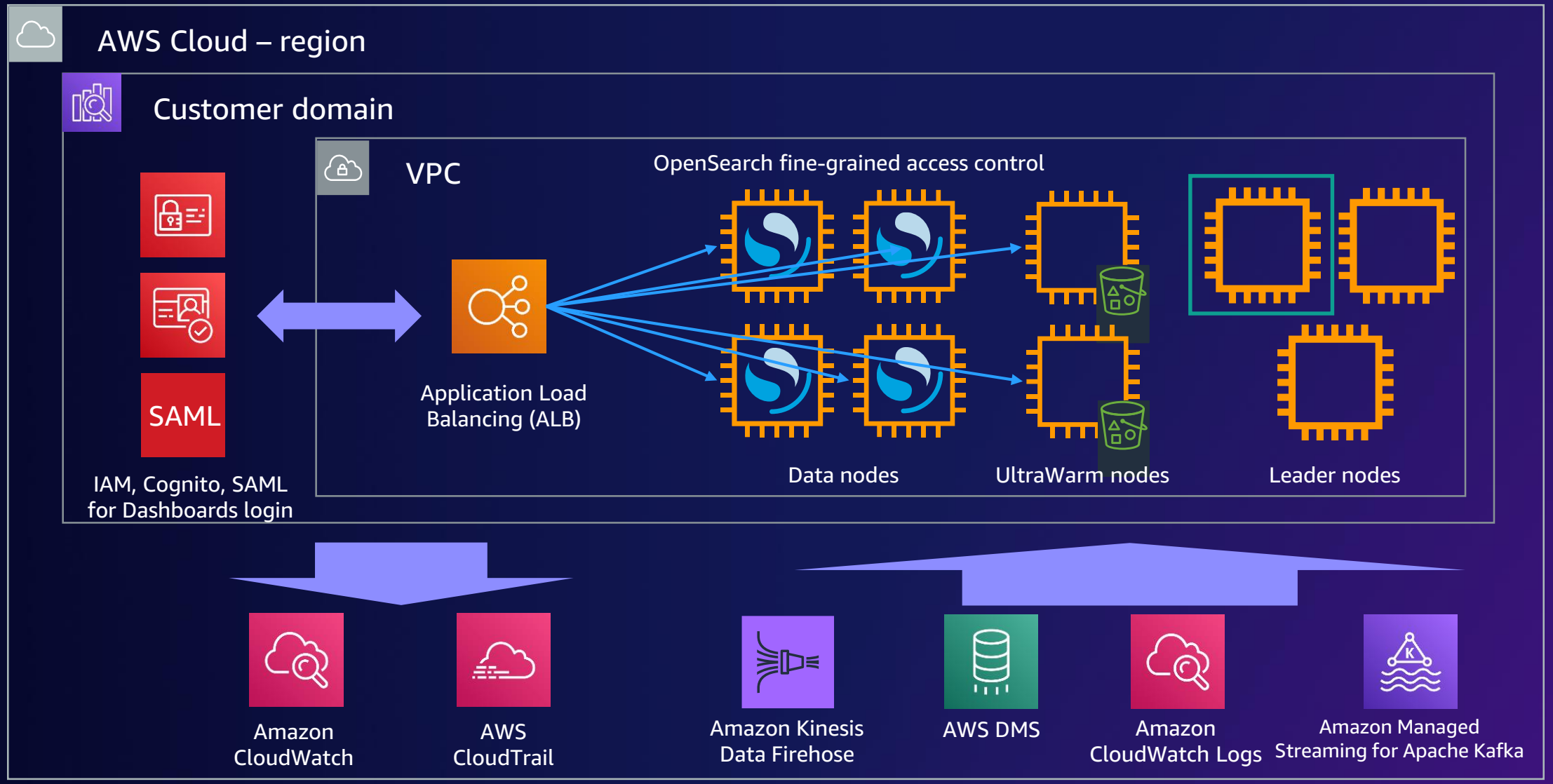
100 * 1.1 * 335 = 36.85 TiB

3xM6g.xlarge hot (min for storage)

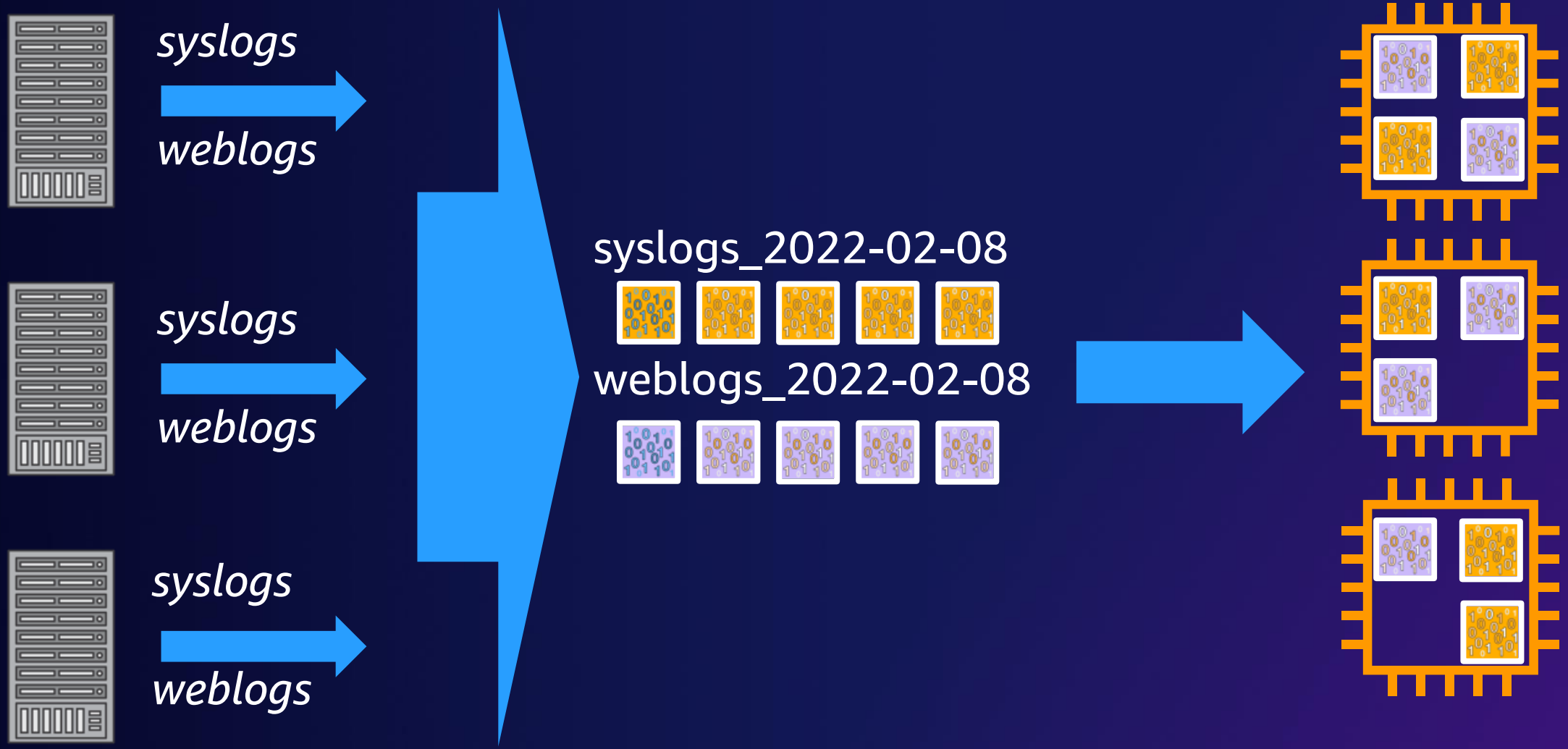
2xUltraWarm.medium

Request processing

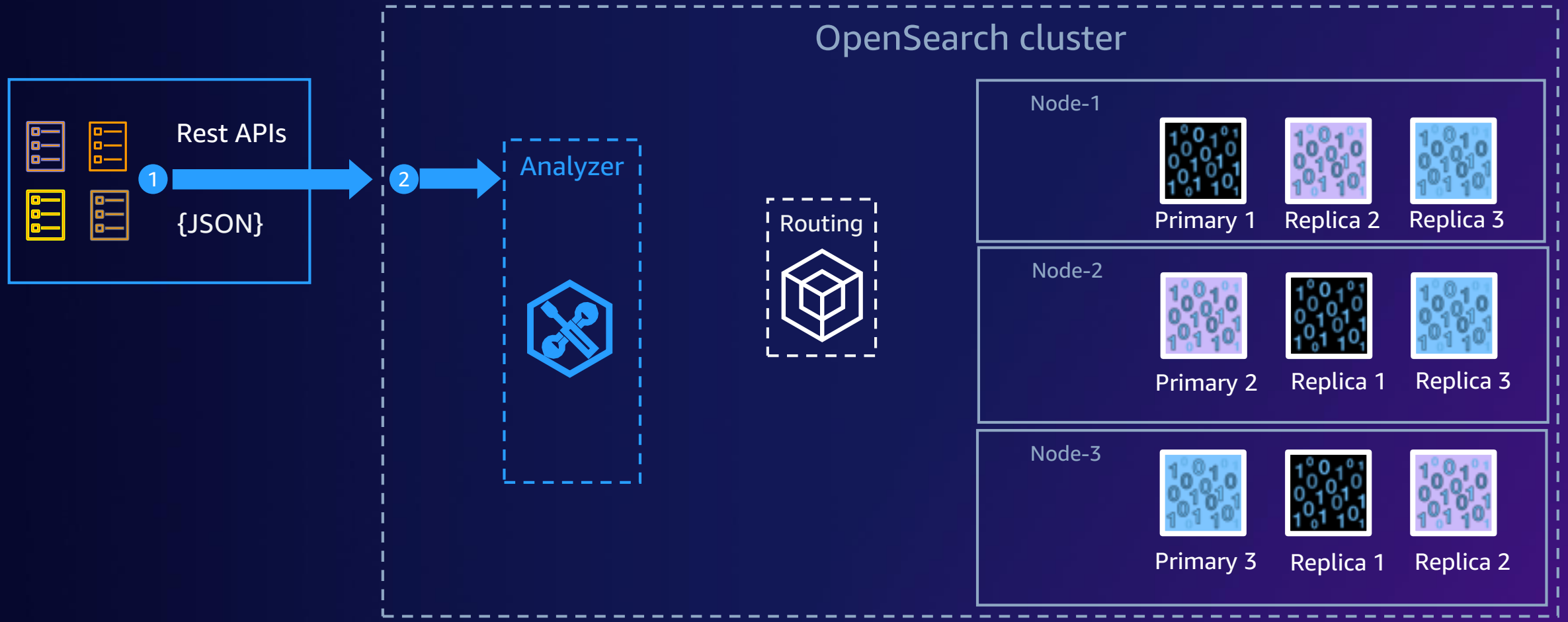
Amazon OpenSearch Service architecture



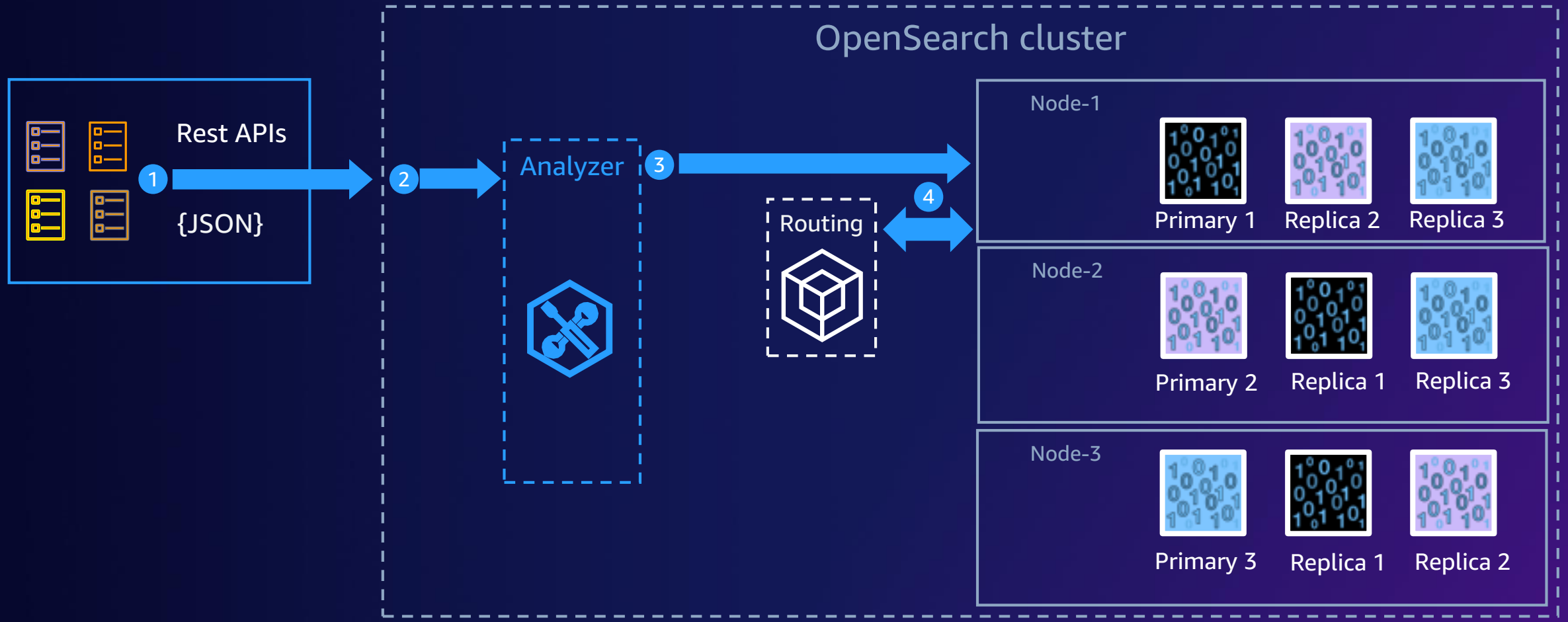
Indexes are distributed via shards (partitions)



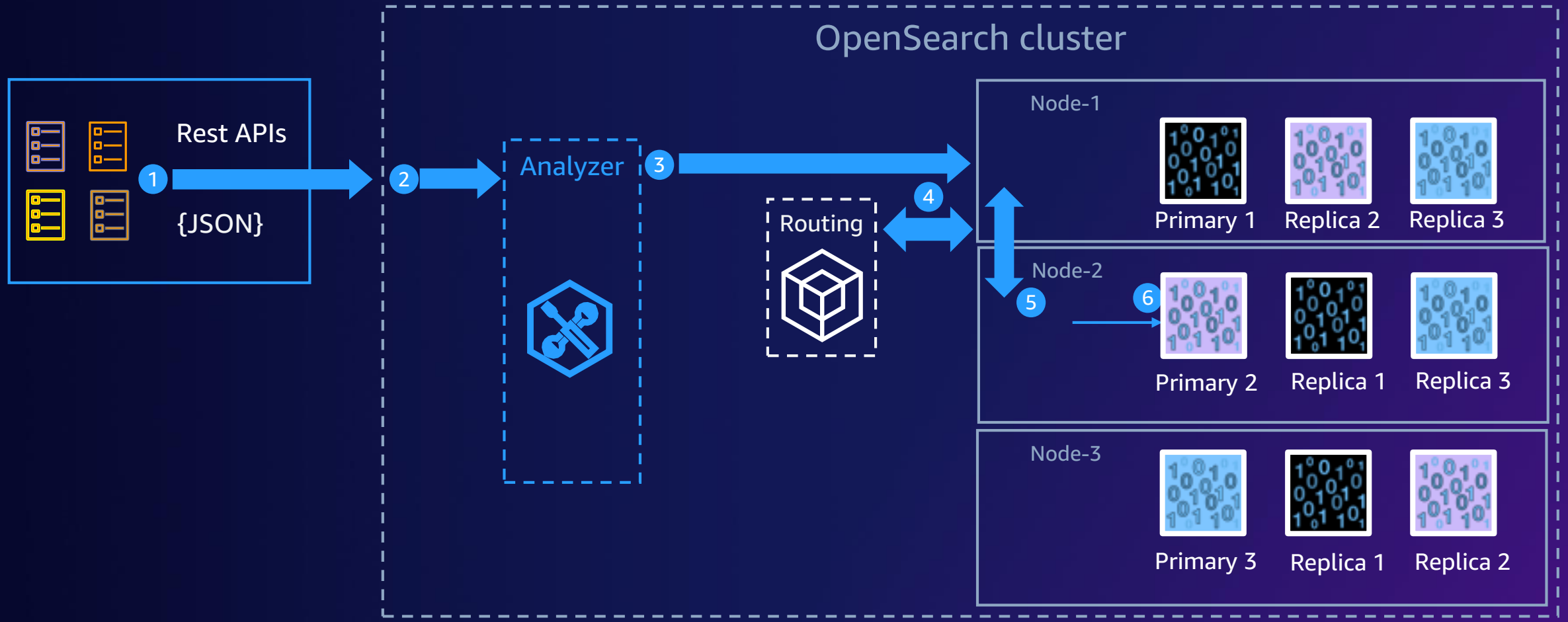
Shards are workers: indexing



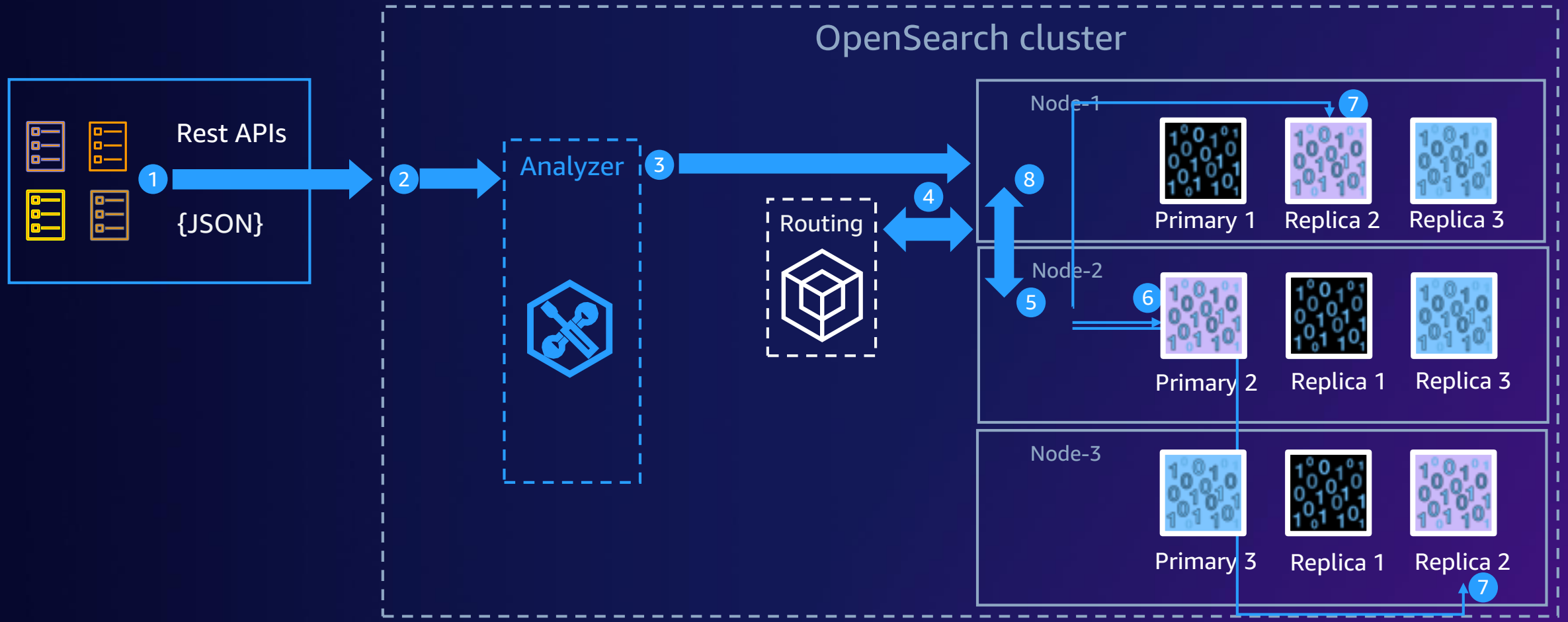
Shards are workers: Indexing



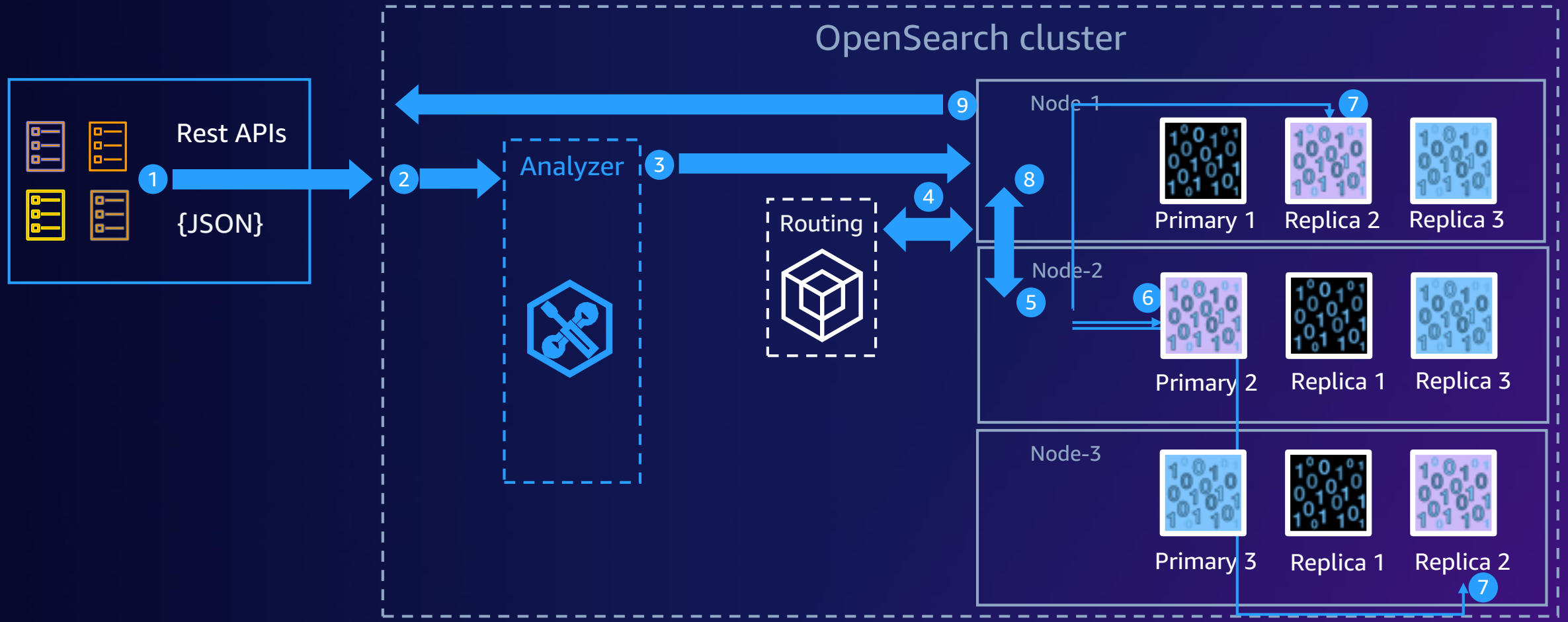
Shards are workers: Indexing



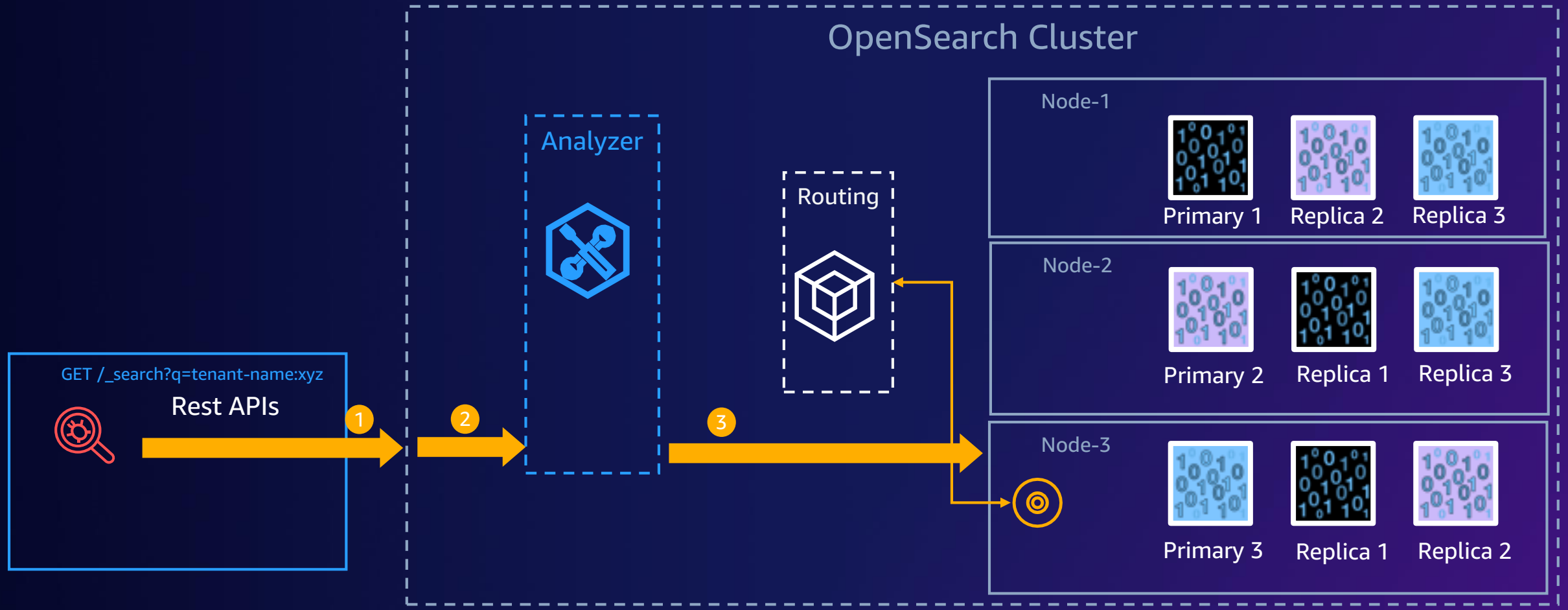
Shards are workers: Indexing



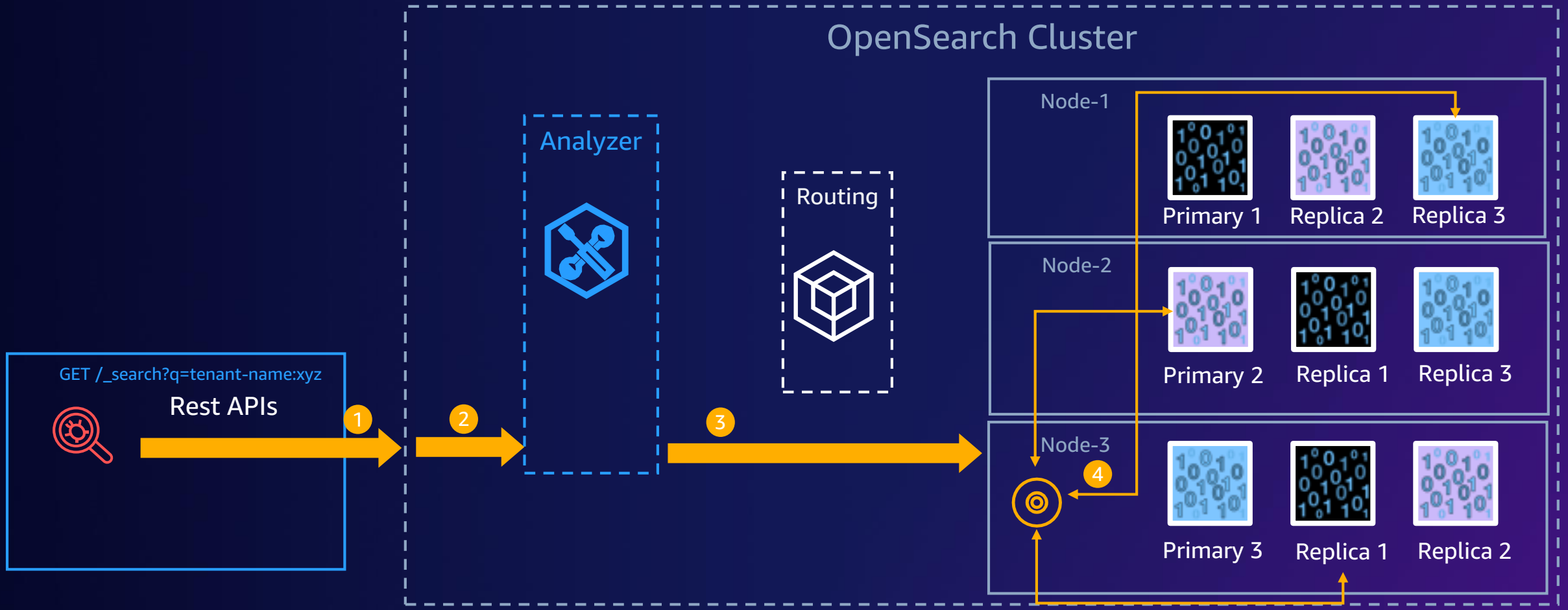
Shards are workers: Indexing



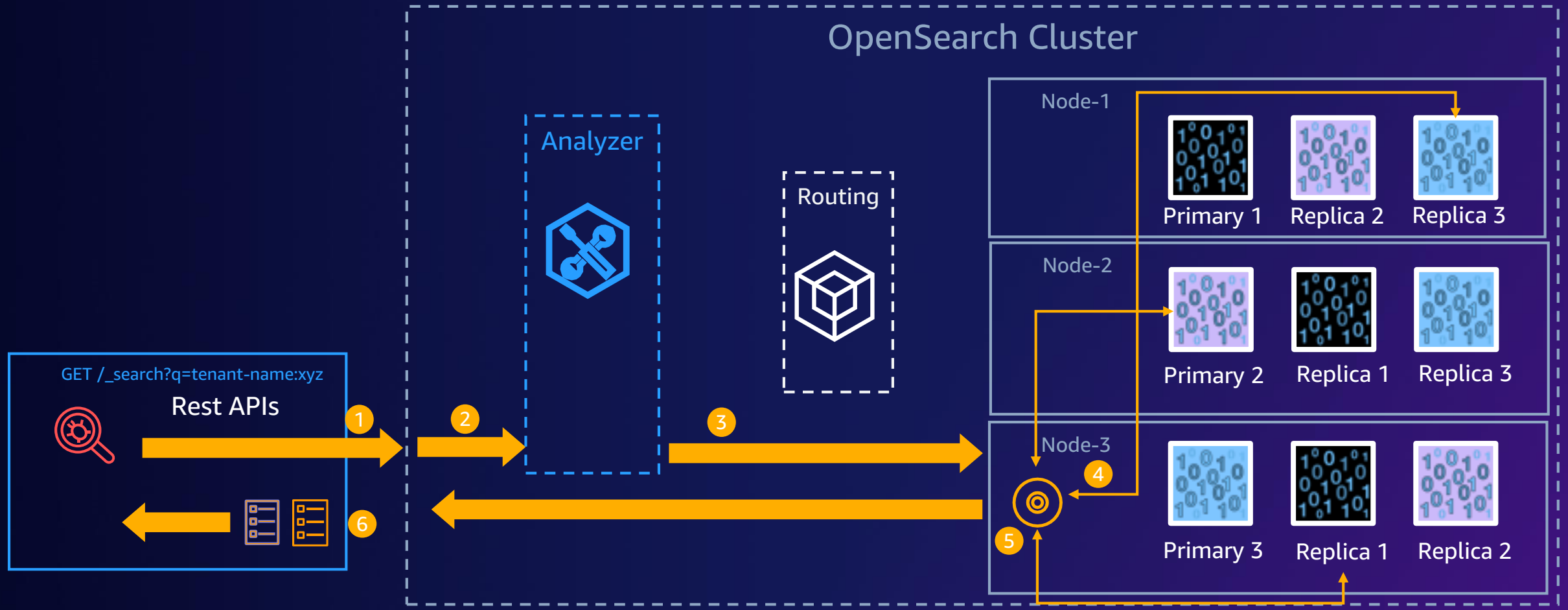
Shards are workers: Query



Shards are workers: Query

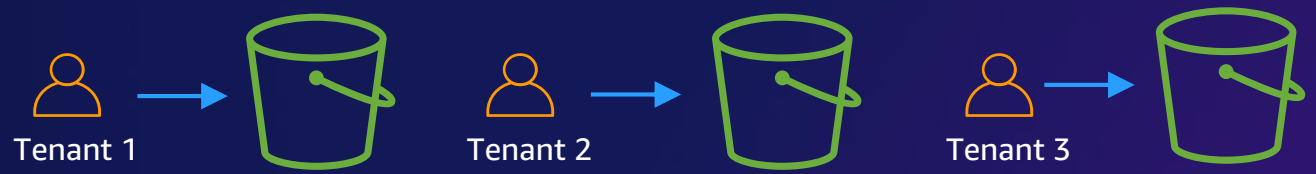


Shards are workers: Query



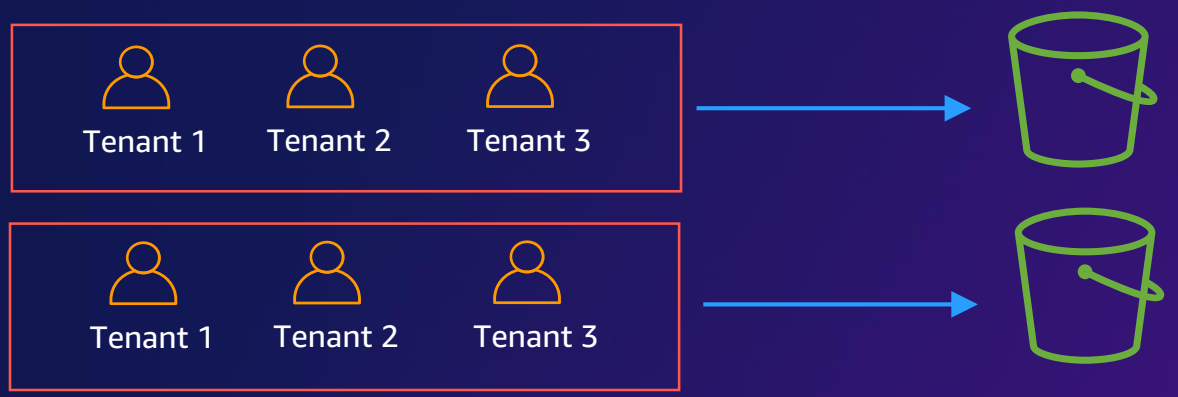
Different tenancy strategies

Siloed strategy



Pooled strategy

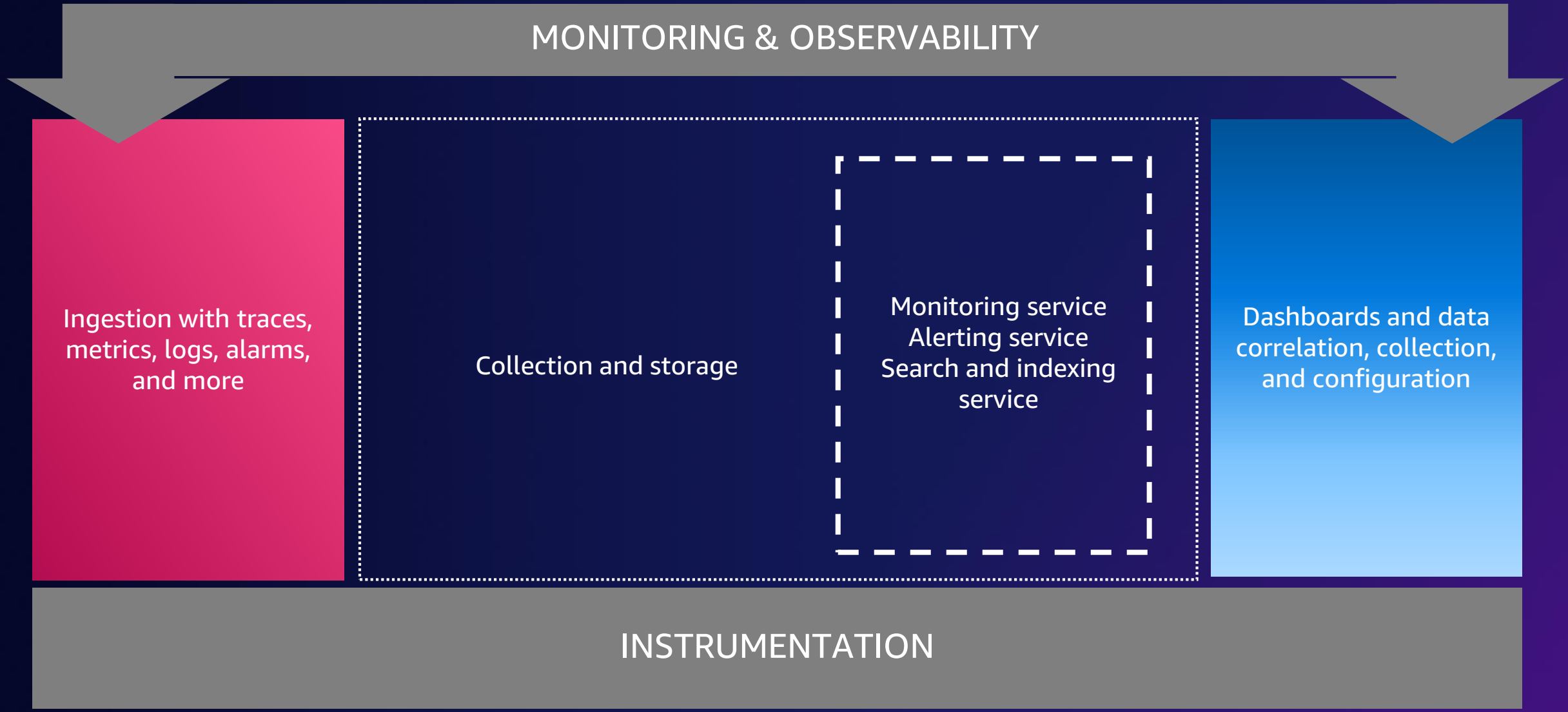
Hybrid strategy



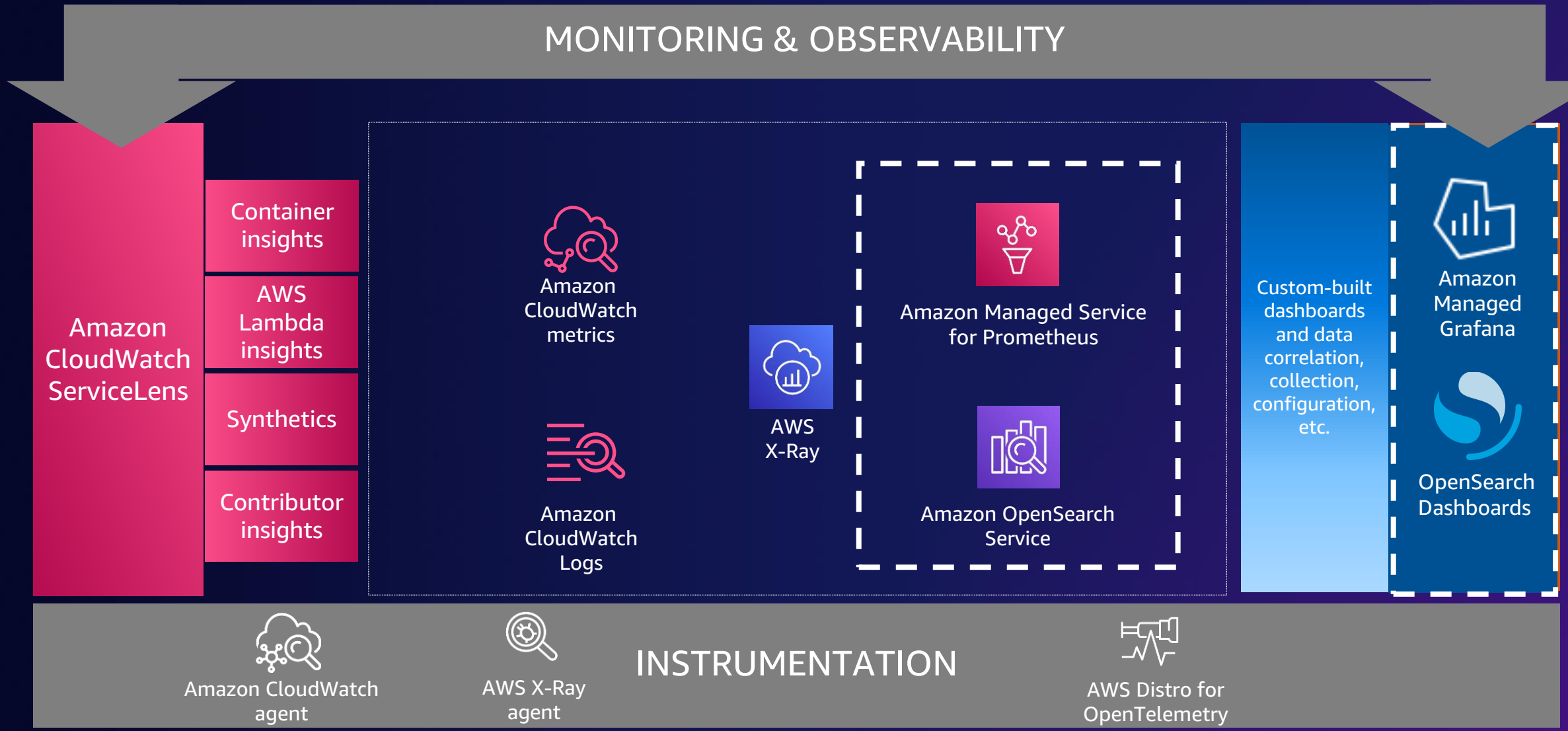
Tenants are isolated based on tenant characteristics

Observability in Amazon OpenSearch Service

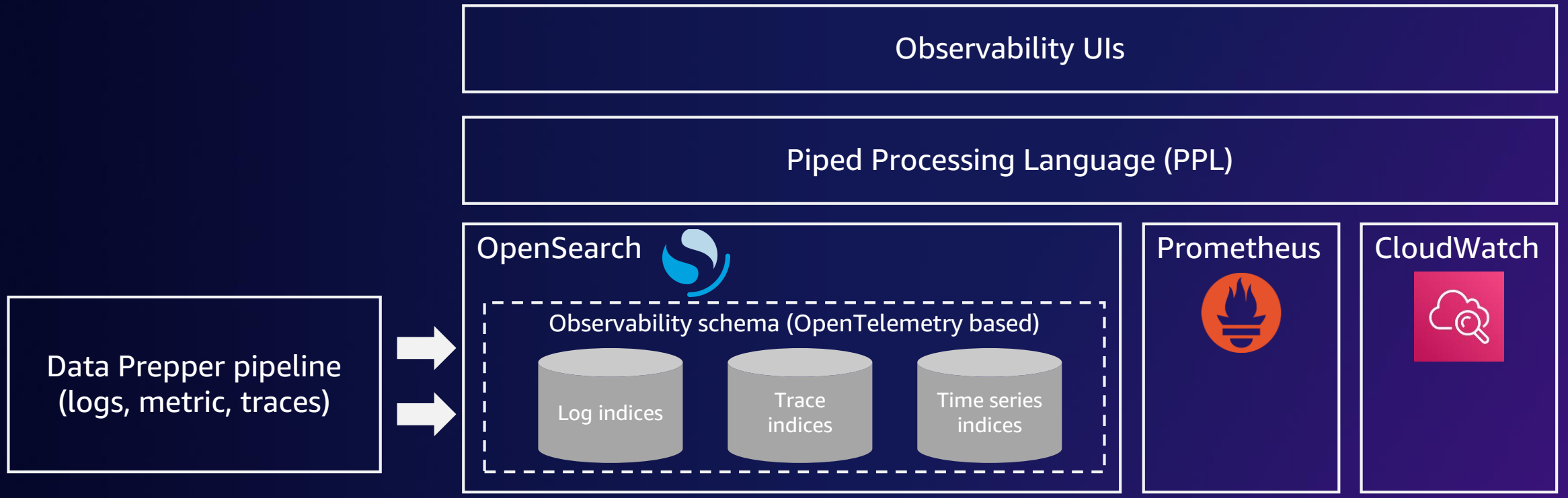
The observability model: Overview



The observability model: Details



Observability implementation overview

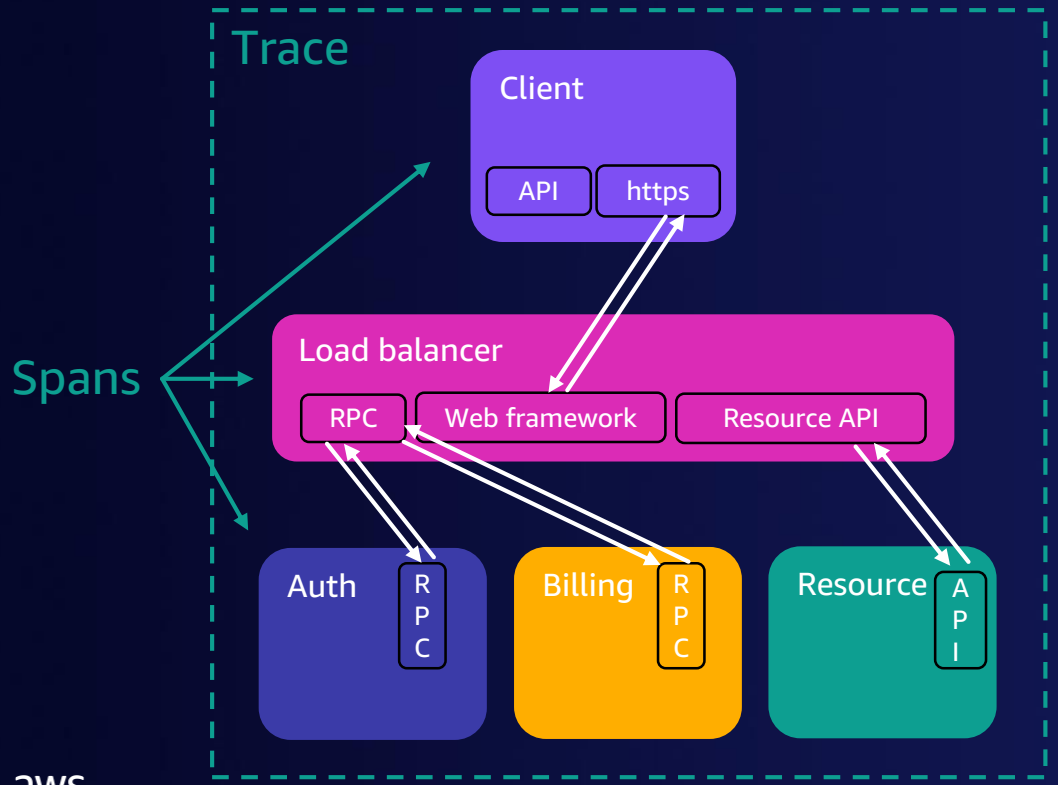


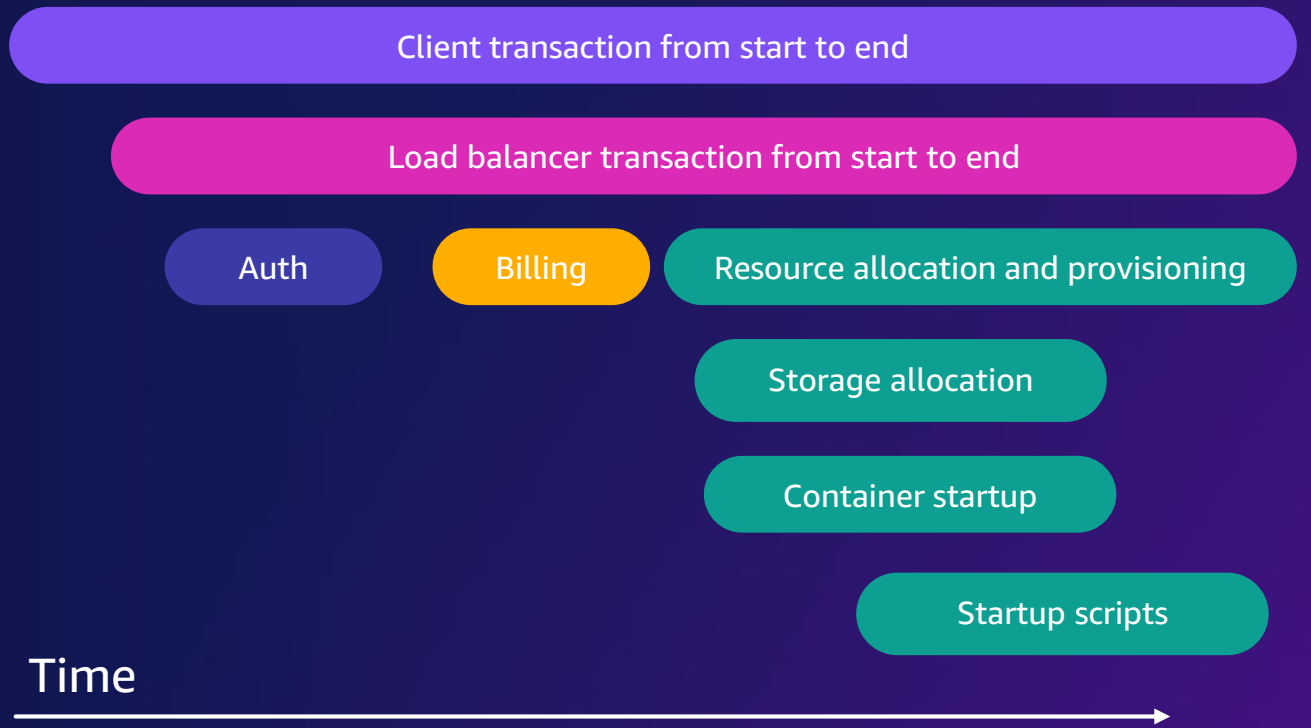
What is distributed tracing?

IDENTIFYING PROBLEMS IN CLOUD APPLICATIONS

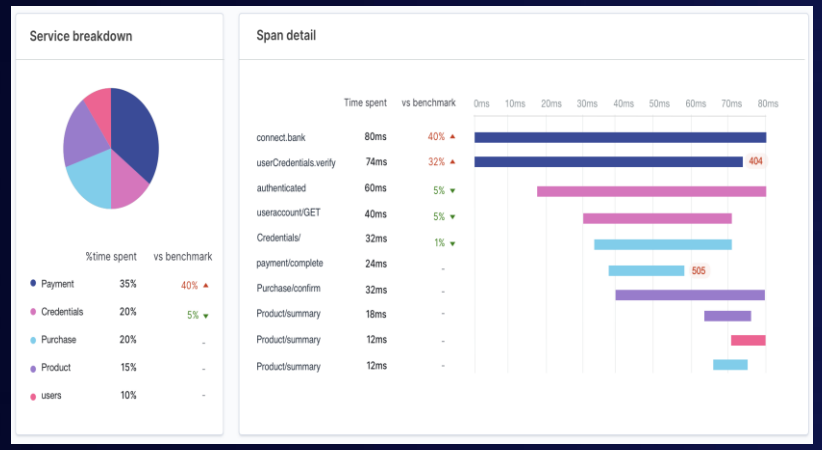
A method of observing requests as they propagate through distributed systems

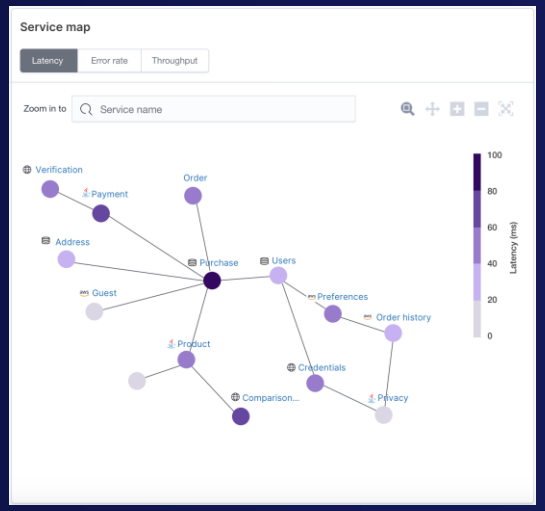
Trace: Hierarchical, end-to-end record of processing a request

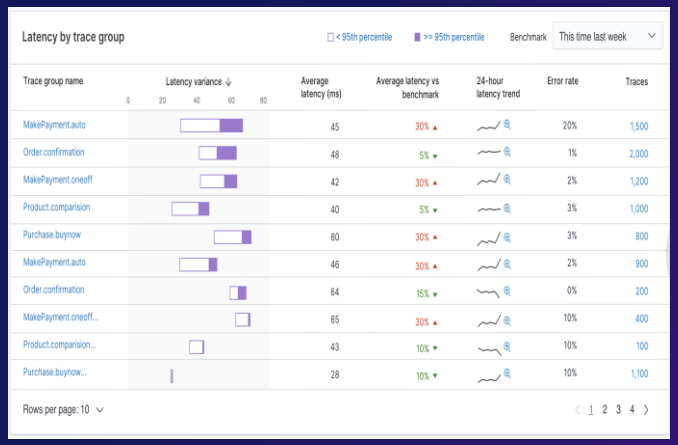




Trace analytics







Trace-span details

- Single request performance
- Diagnose root cause

Service maps

- End-to-end view
- Isolate issues to services

Trace groups

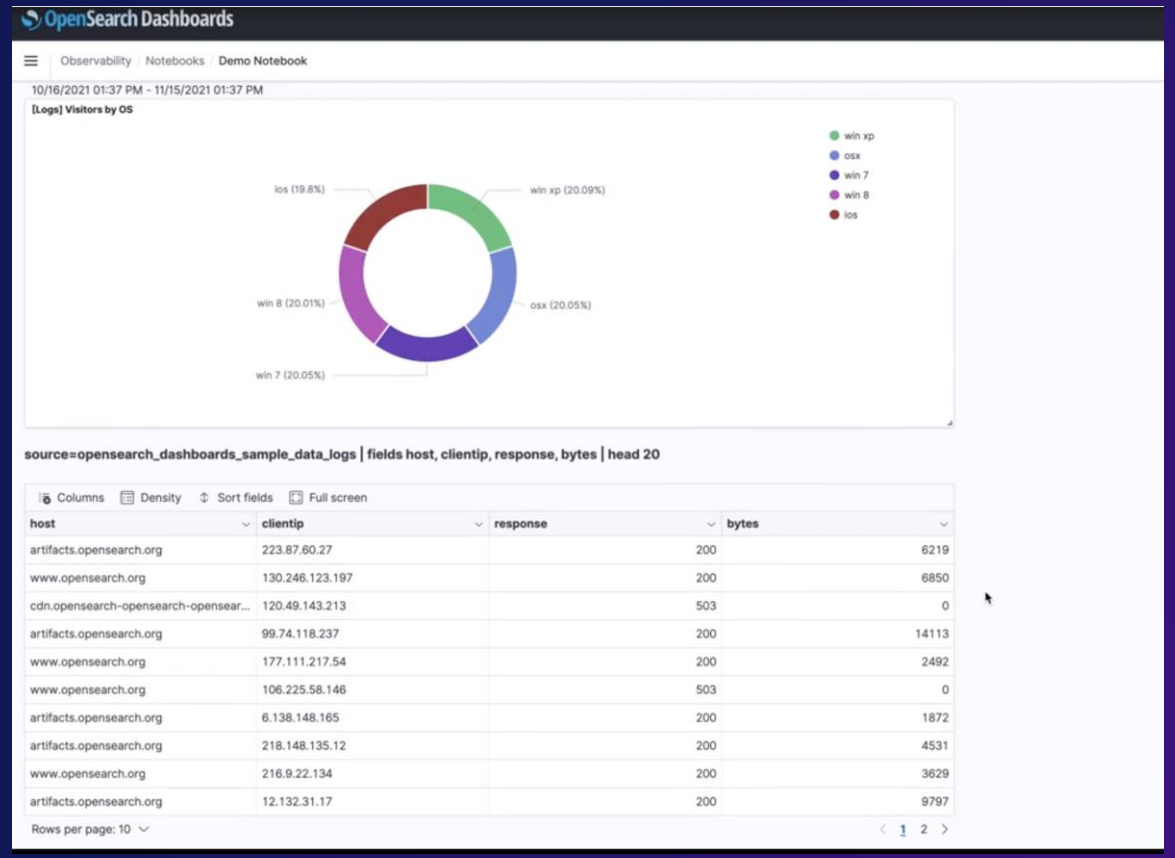
- Monitor performance
- Identify issues early

Storytelling with notebooks

A notebook is a document made up of cells or paragraphs that can combine markdown, SQL/PPL queries, and visualizations

Support for multi-timelines so that users can easily tell a story

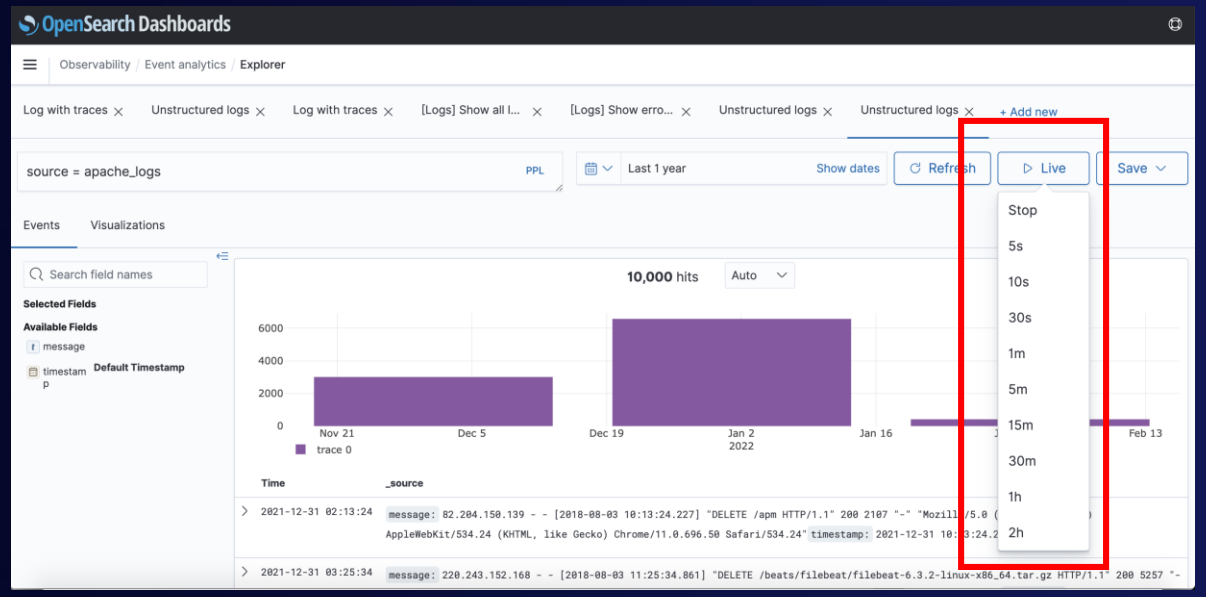
Can be shared as an OpenSearch Dashboards link, PDF, or PNG



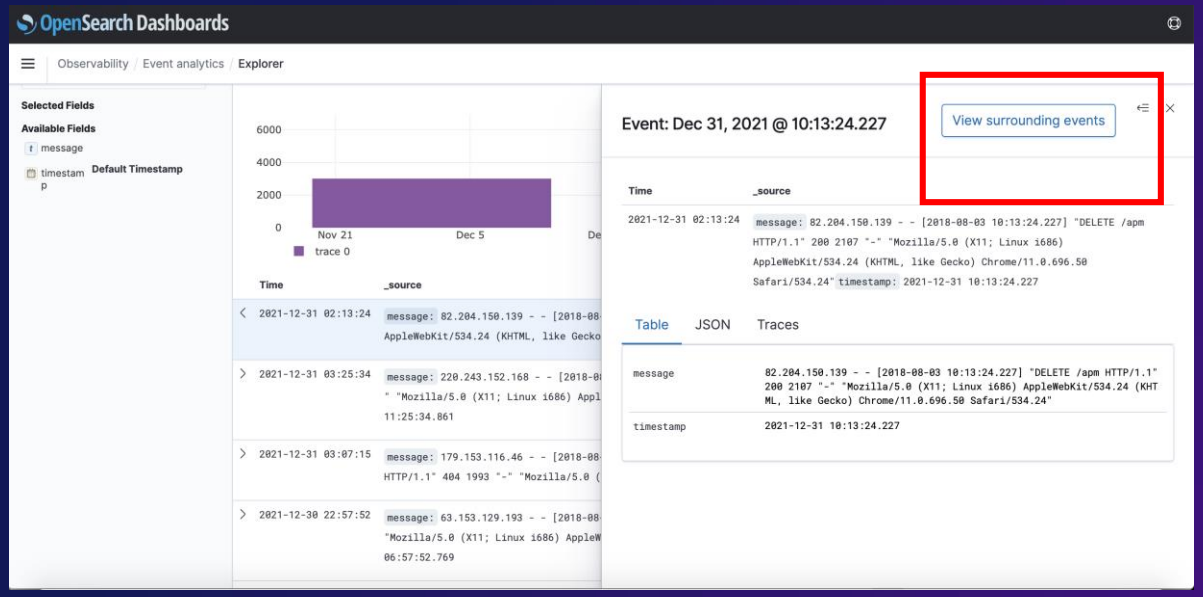
- Common use cases include creating postmortem reports, designing runbooks, building live infrastructure reports, or even documentation

Log monitoring

Live tail



Log surround



Customer examples

Amazon OpenSearch Service customers

Software and internet

Adobe · IACI · AUTODESK

Education technology

McGraw Hill Education · INSTRUCTURE · Blackboard

BioTech and pharma

MONSANTO · Bristol-Myers Squibb

Financial services

THOMSON REUTERS · stripe · KICKSTARTER

Media and entertainment

mlbam · NETFLIX · FOX

Social media

cookpad · ancestry · Nextdoor

Telecommunications

Jio · COMCAST · T-Mobile

Travel and transportation

lyft · Expedia · UBER

Real estate

Zillow · move

Logistics and operations

here · elementum · infor

Publishing

FT FINANCIAL TIMES · ELSEVIER · The Washington Post

Other

Canon · SAMSUNG · British Gas

Use case #1

Observability

Pinterest case study

(https://go.aws/3wqGToQ)

Why observability?

You need to correlate logs, metrics, and traces to gain insights into application health and performance and resolve issues across the business

How Amazon OpenSearch Service can help

Centralizes log analytics to identify or predict performance problems across your business. With cross-cluster search, you can analyze and query all of your log data via a single OpenSearch Dashboards interface

Use case #2

Application & infrastructure monitoring

Why application and infrastructure monitoring?

You need to proactively monitor your applications and infrastructure log data to find performance issues faster and improve operational health

How Amazon OpenSearch Service can help

Provides real-time search and log analytics capabilities to identify or predict performance problems and enable your teams to do real-time root cause and forensic analysis, therefore reducing mean time to detect (MTTD) and mean time to resolve (MTTR) issues

AUTODESK

Autodesk case study

(https://go.aws/3cmnHSu)

Use case #3

Search

COMPASS

Compass blog

(https://go.aws/3CuNB0O)

Why search?

You need a fast search experience for your applications, websites, and data lake catalogs, allowing your users to quickly find relevant data

How Amazon OpenSearch Service can help

Delivers high-quality and personalized search results to customers. You get access to all of Elasticsearch's search APIs, supporting natural language search, auto-completion, faceted search, adjustable ranking, and location-aware search

Use case #4

Security monitoring

Pearson case study

(https://go.aws/3QTINqg)

Why security monitoring?

You need to keep your data safe, preventing security threats such as data breaches, unauthorized login attempts, DoS attacks, and fraud

How Amazon OpenSearch Service can help

Accelerate security incident detection, forensic analysis, and response by being able to quickly analyze logs from disparate applications and systems across your network

Thank you!

Jon Handler
handler@amazon.com

Muthu Pitchaimani
muthupmk@amazon.com