



# Monitoring and Observability for Modernized Applications

Viji Sarathy

Specialist Solutions Architect, Containers



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

# Agenda

- Observability Best Practices
- AWS Observability Tools
- Open Source Observability Tools

# What is Observability?



A measure of how well we can understand a system from the work it does

"90% of the methods in this service complete in under 200 milliseconds"

"This API is handling 203HTTP requests per second"

"CPU utilization for this service is at 85%"

# Observability matters because ...



---

Visibility



---

Real-time  
troubleshooting



---

Customer  
experience



---

Applications = \$\$

**Operational**

**Business**

# What is Instrumentation?



“Calls to this database took, on average, took 50 milliseconds”

**Instrumentation:** measuring events in software using code  
(a type of white-box monitoring)

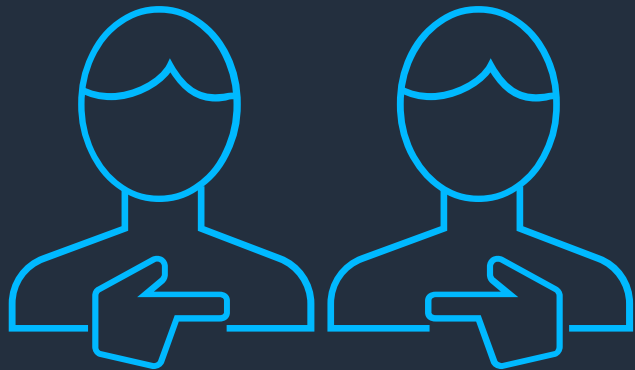
# ***Good data* can help with the technical shift to new systems**



## **Technical**

- Improved debugging and troubleshooting
- Designs validated with data
- Reduced defects; more issues caught proactively
- Improved feature velocity

# ***Good data* can help with the cultural shift to new systems**



## **Cultural**

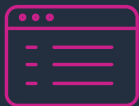
- Builds transparency across teams
- Shared understanding of complex components
- Decisions not (entirely) driven or explained by gut feelings or guessing
- Freedom to experiment
- Blameless culture
- Context not control

But...

How do we make  
microservices and serverless functions  
**observable?**



# #1: Observable systems should emit events: Metrics, logs, and traces



Logs

"The database won't start after the update"



Metrics

"Our application is 35% slower than last week  
after this configuration change"

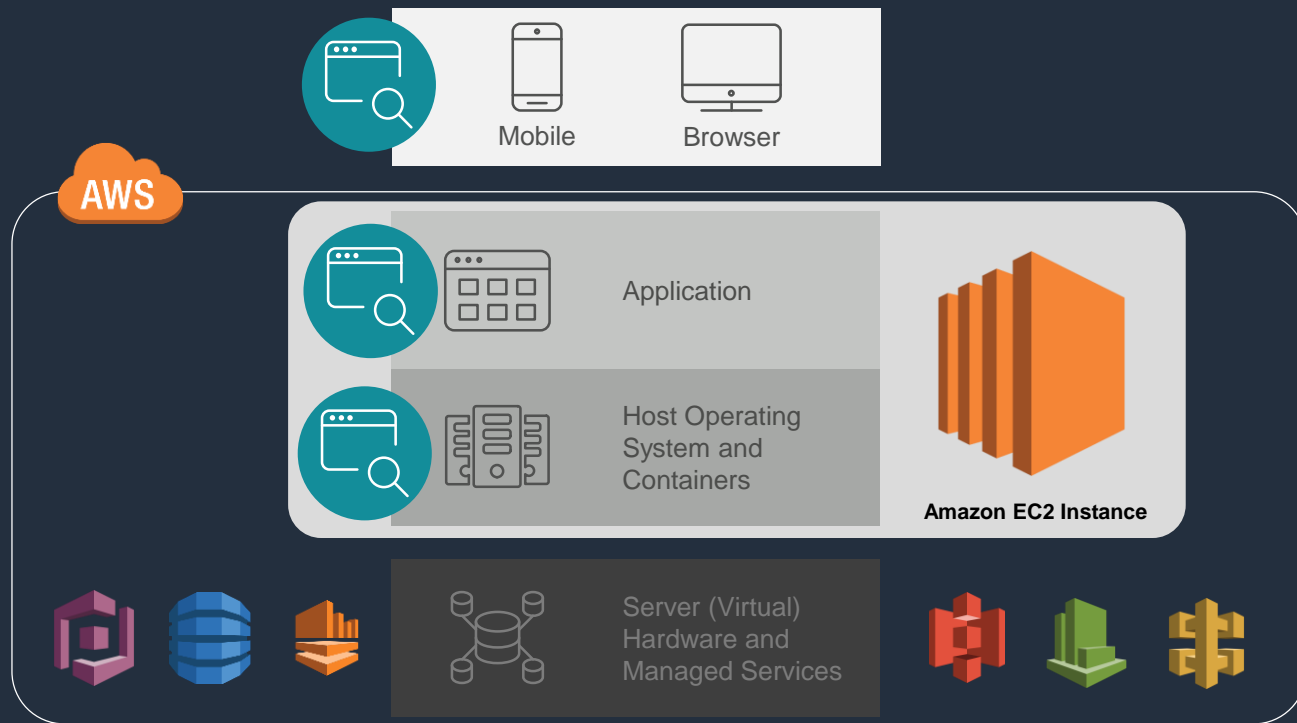


Traces

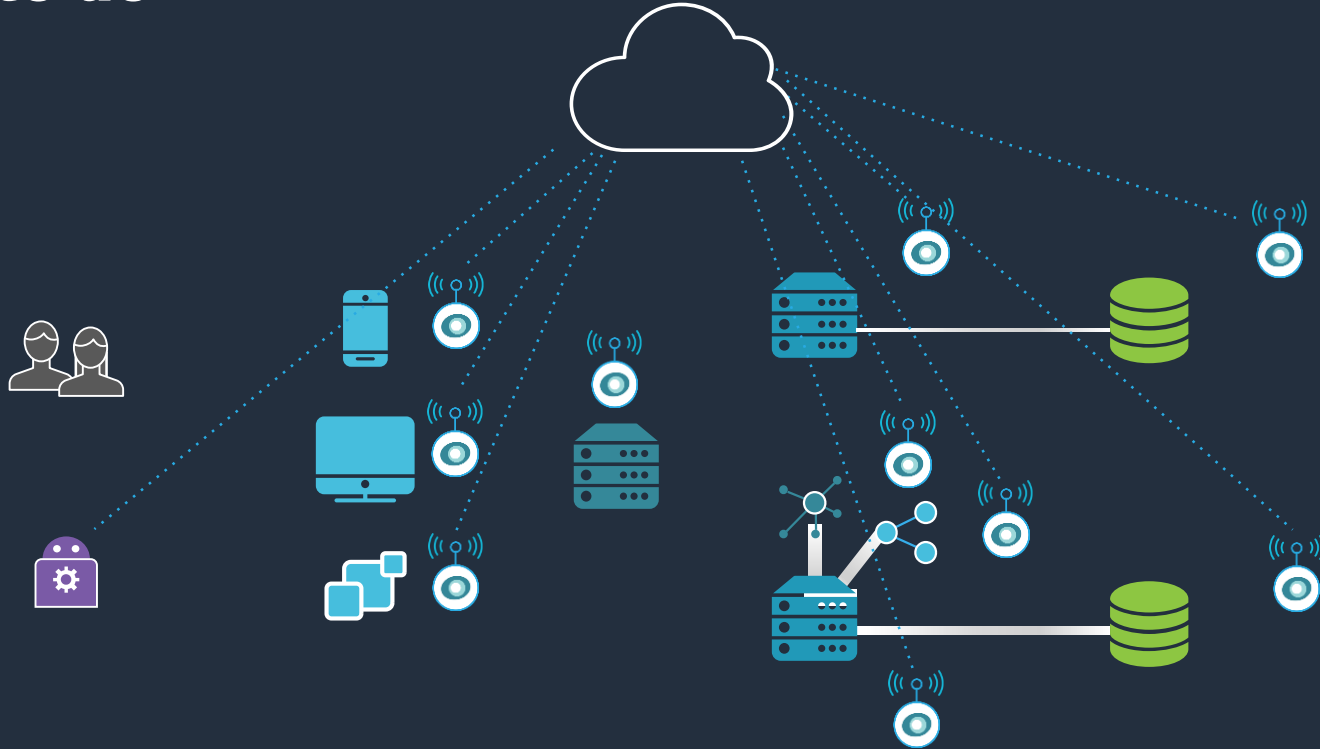
"What are the dependencies for this service?"



## #2: All components should be instrumented



# #3: Instrumentation should not be opt-in, manual, or hard to do



# AWS Observability tools

**What:** Immutable, timestamped record of discrete events that happened over time

**Why:** Useful for uncovering emergent and unpredictable behavior



CloudWatch Logs

AWS observability tools



CloudWatch



traces

**What:** Representation of a series of related distributed events that encode the end-to-end request flow through a distributed system

**Why:** Provides visibility into both the path traversed by a request as well as the structure of a request

**What:** Numeric representation of data measured over intervals of time

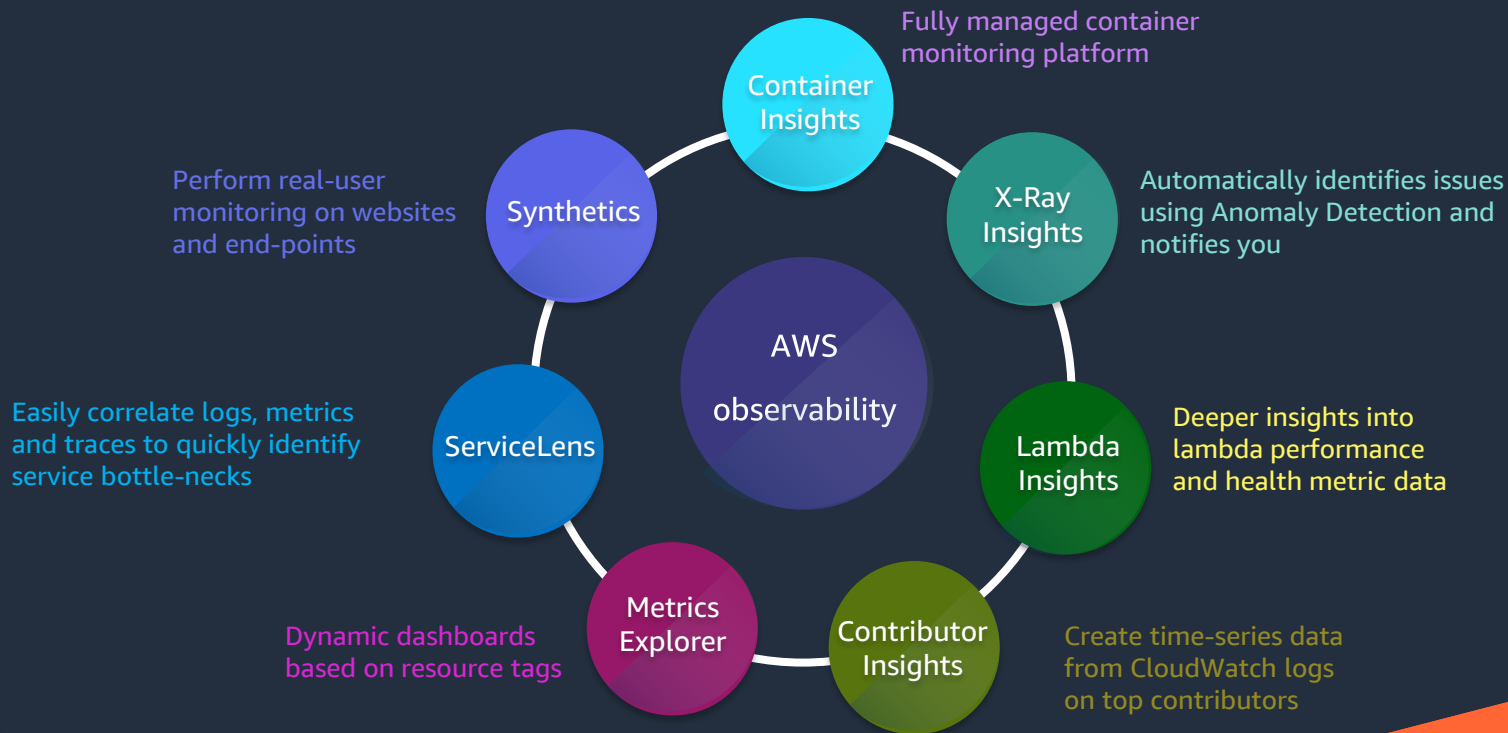
**Why:** Useful for identifying trends, mathematical modeling, and prediction

# AWS Observability tools

► Infrastructure monitoring

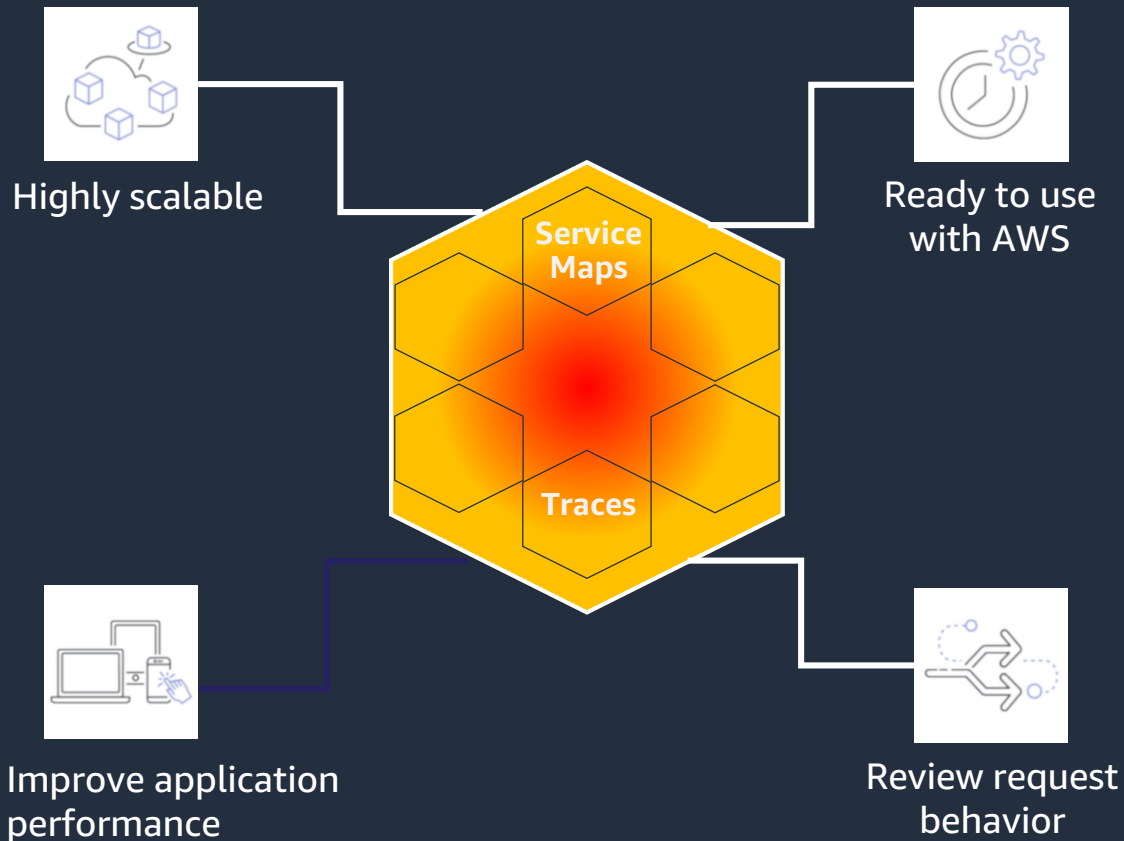
► Application monitoring

► Synthetic monitoring



# AWS X-Ray

Analyze and debug  
production,  
distributed  
applications

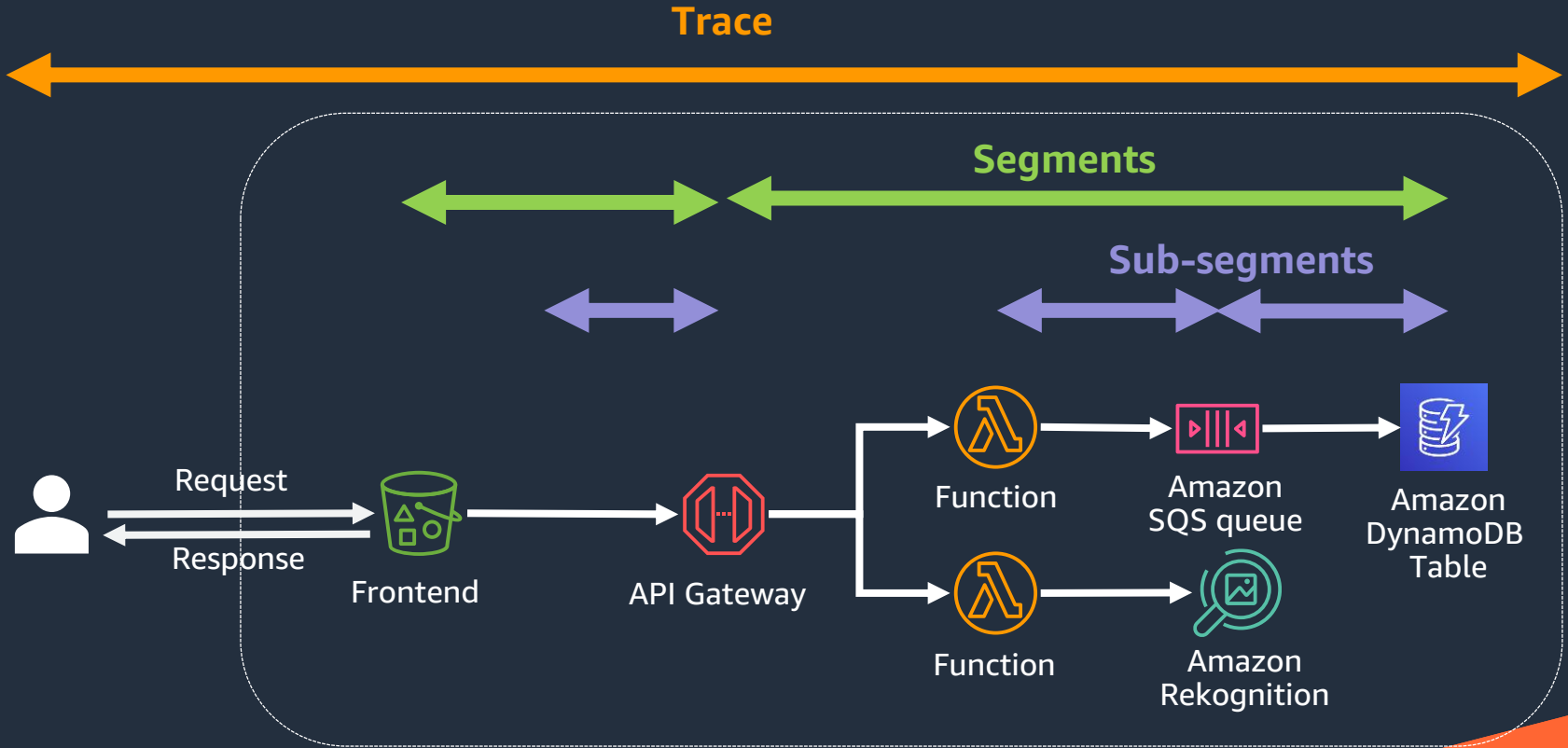


# How does AWS X-Ray help?

- Analyze and debug performance of your distributed applications.
- View latency distribution and pinpoint performance bottlenecks.
- Identify specific user impact across your applications.
- Works across different AWS and non-AWS services.
- Ready to use in production with low latency in real-time.

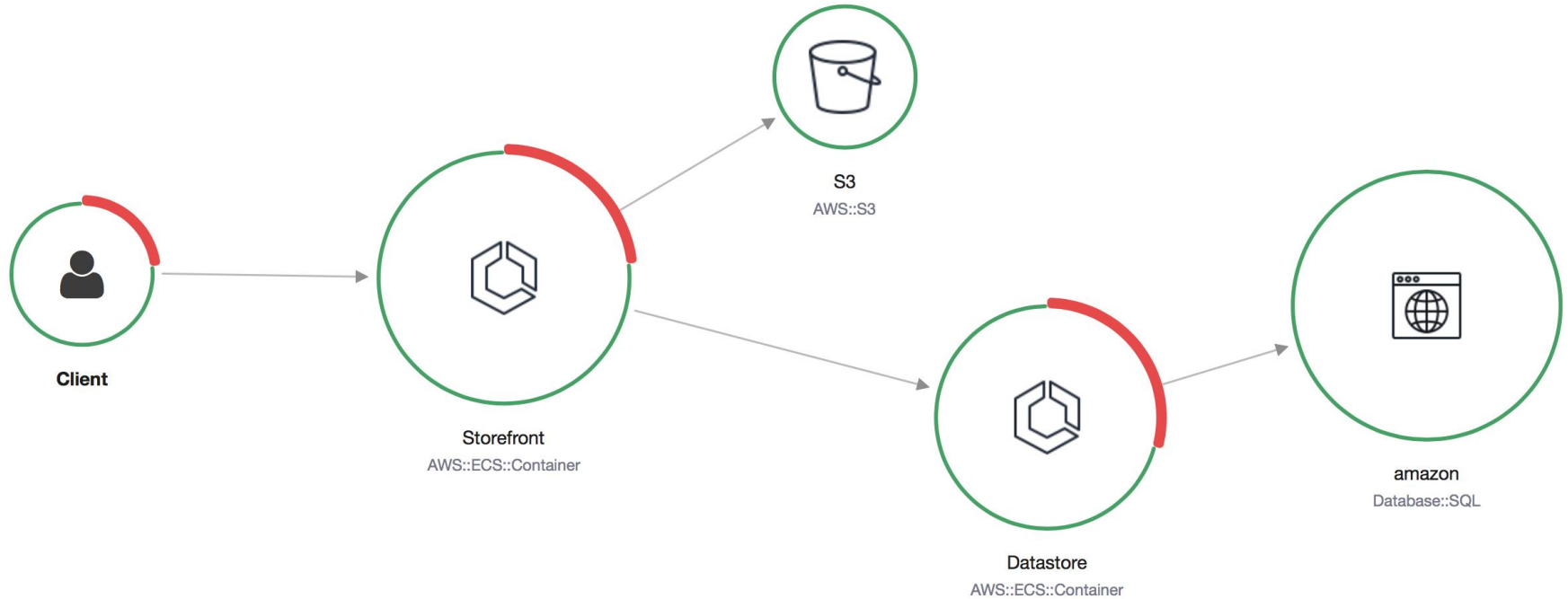
Enables you to get started quickly without having to manually instrument your application code to log metadata about requests

# AWS X-Ray concepts

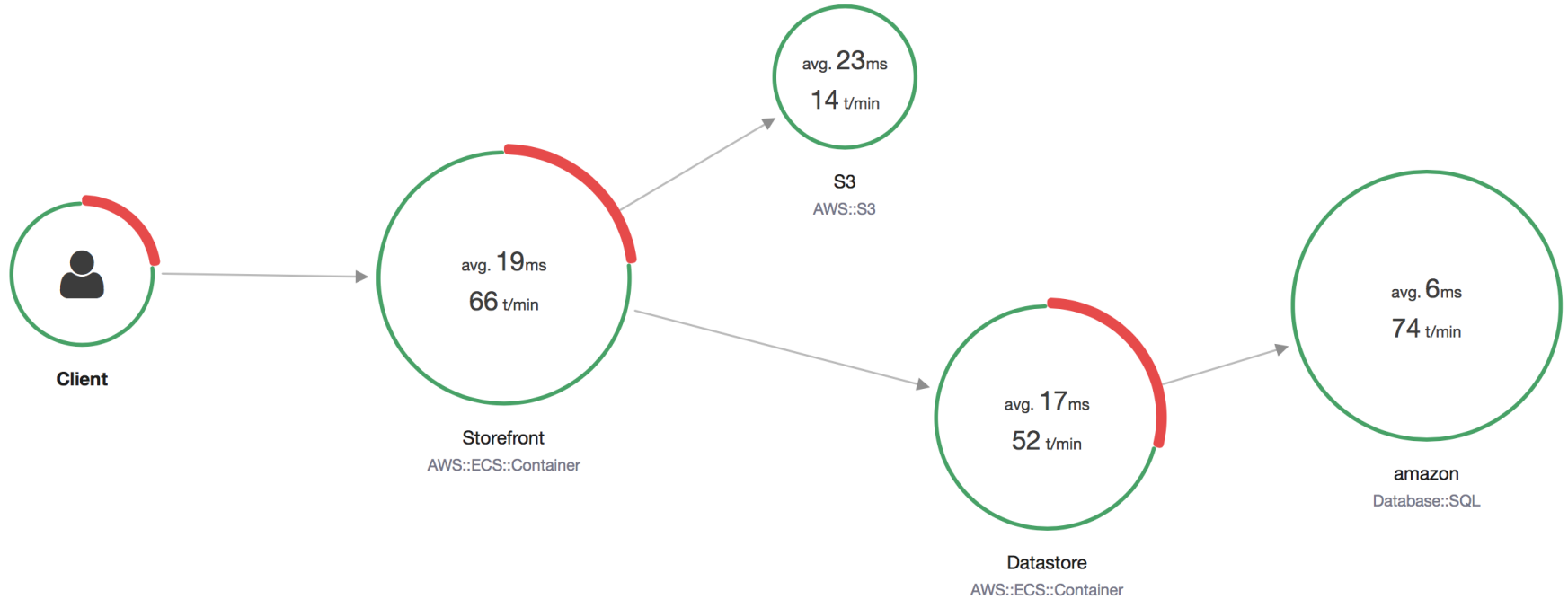




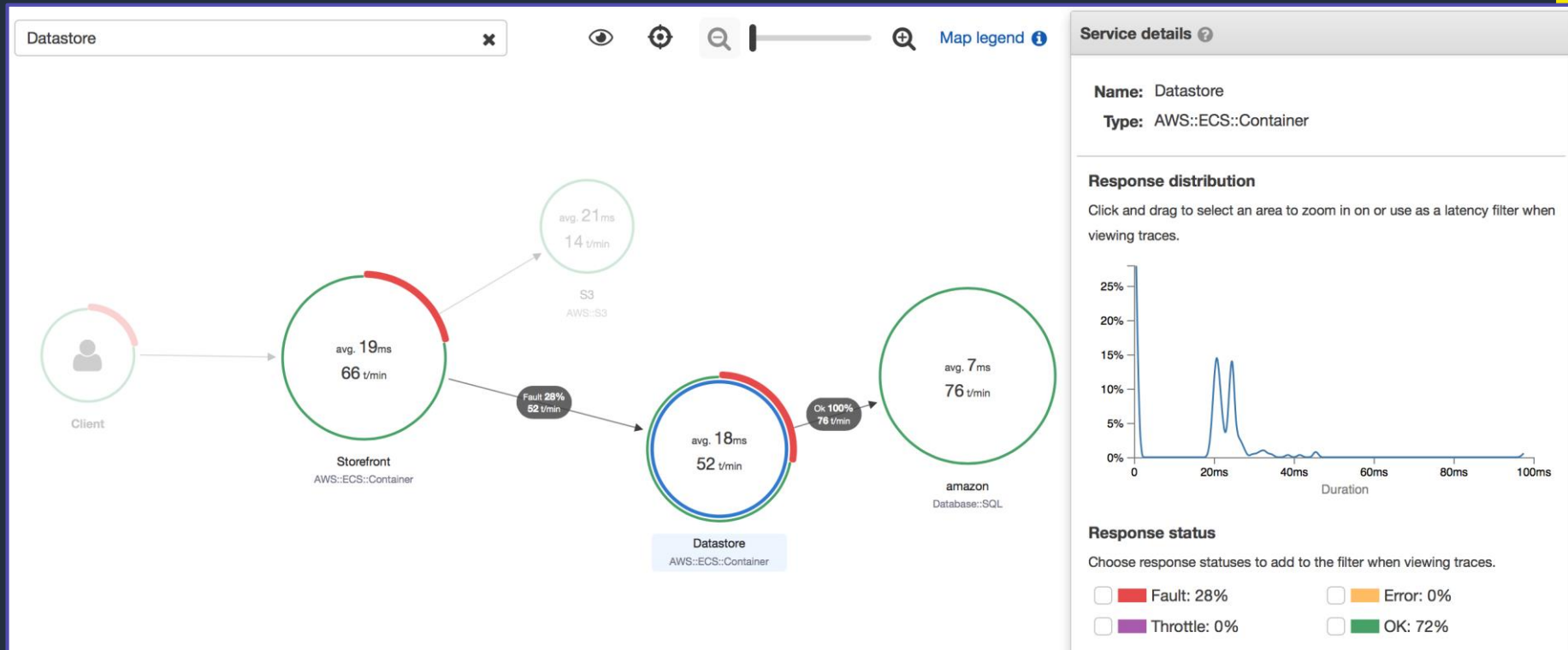
# Visualize service graph



# Visualize service graph



# Identify performance bottlenecks



# X-Ray SDK

Available for Java, .NET, .NET Core, Ruby, Python, Go, and Node.js

Adds filters to automatically capture metadata for calls to

- AWS services using the AWS SDK
- Non-AWS services over HTTP and HTTPS (third-party APIs)
- Databases (MySQL, PostgreSQL, and Amazon DynamoDB)
- Queues (Amazon SQS)

# AWS X-Ray for Lambda

X-Ray agent is natively built into Lambda

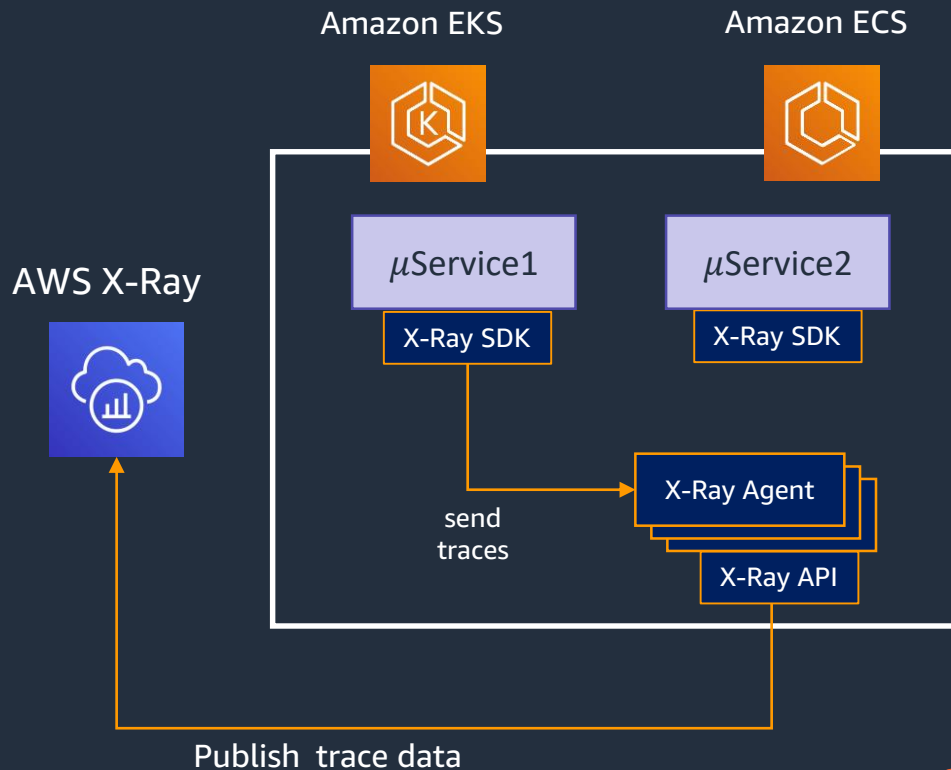
Identify initialization and cold starts in Lambda

Pinpoint issues in downstream services called from your Lambda function

Happens with low latency in real time; can see traces in seconds

# AWS X-Ray for Amazon ECS/EKS

- Microservices instrumented with X-Ray SDK send **segment** data to X-Ray agent in the cluster
- X-Ray agent buffers segments in a queue and uploads them to X-Ray in batches
- X-Ray groups segments that have a common request into **traces** which are used to generate a **service graph** that provides a visual representation of your application



# CloudWatch metrics for Lambda

- Runtime metrics for Lambda functions are available in CloudWatch across three different categories
- **Invocation Metrics**
  - **Invocations**
  - **Errors**
  - **DeadLetterErrors**
  - **Throttles**

# CloudWatch metrics for Lambda

- Runtime metrics for Lambda functions are available in CloudWatch across three different categories
  - **Duration**
  - **IteratorAge**
- Invocation Metrics
- **Performance Metrics**



# CloudWatch metrics for Lambda

- Runtime metrics for Lambda functions are available in CloudWatch across three different categories
  - Invocation Metrics
  - Performance Metrics
  - **Concurrency Metrics**
- **ConcurrentExecutions**
  - **ProvisionedConcurrentExecutions**

# CloudWatch metrics for Serverless

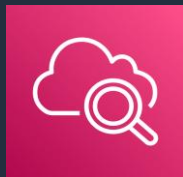
API Gateway	DynamoDB	SQS
Count	Read Throttle Events	Approximate Age Of Oldest Message
Cache Hit Cache Miss	Write Throttle Events	Approximate Number Of Messages Visible
Latency Integration Latency	System Errors	Number Of Messages Sent
4XX Errors 5XX Errors		Number Of Messages Received

# CloudWatch Container Insights

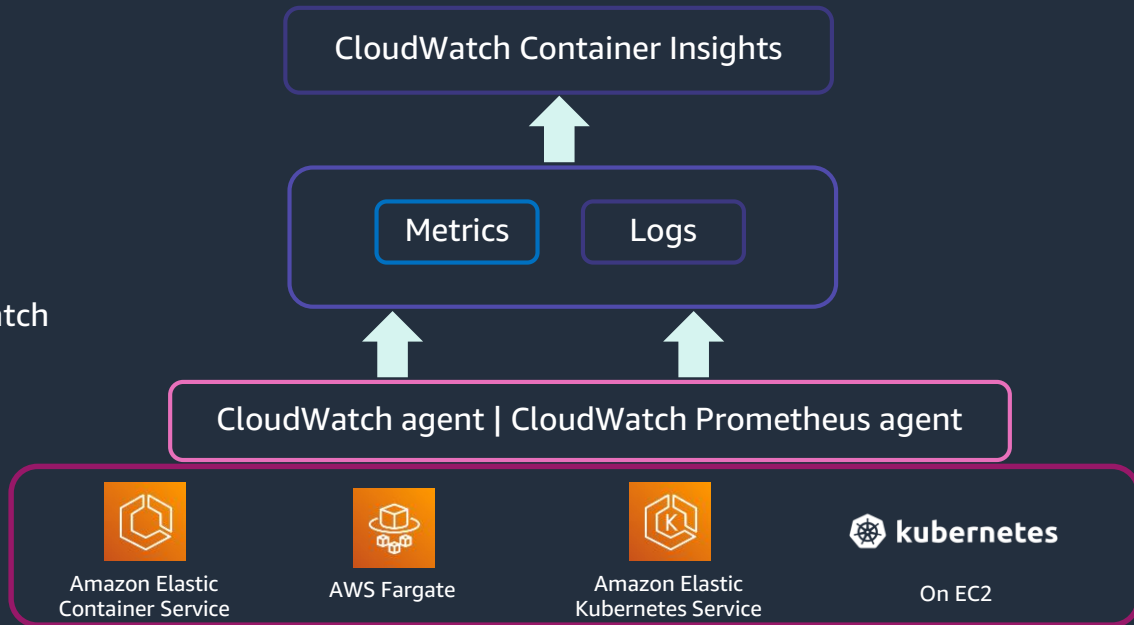
Built-in dashboards to see performance metrics for cluster resources at different levels

Out of the box dashboards for popular workloads such as AppMesh, Java/JMX, NGINX, HAProxy etc

Collect Prometheus metrics from workloads

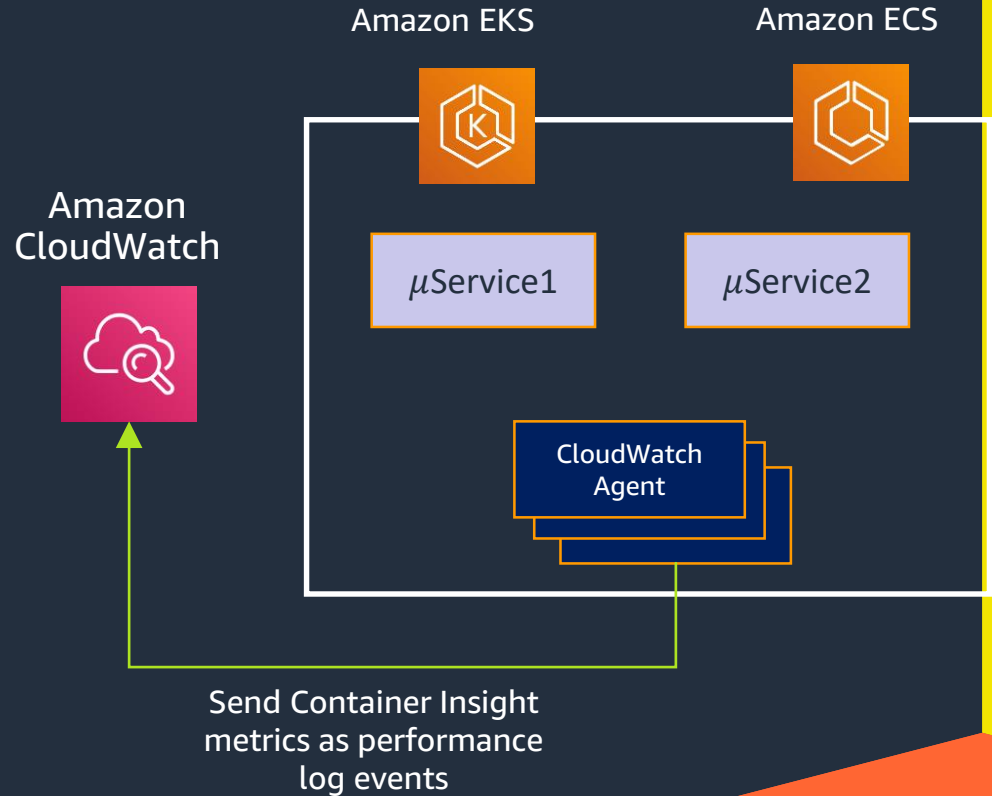


Amazon CloudWatch



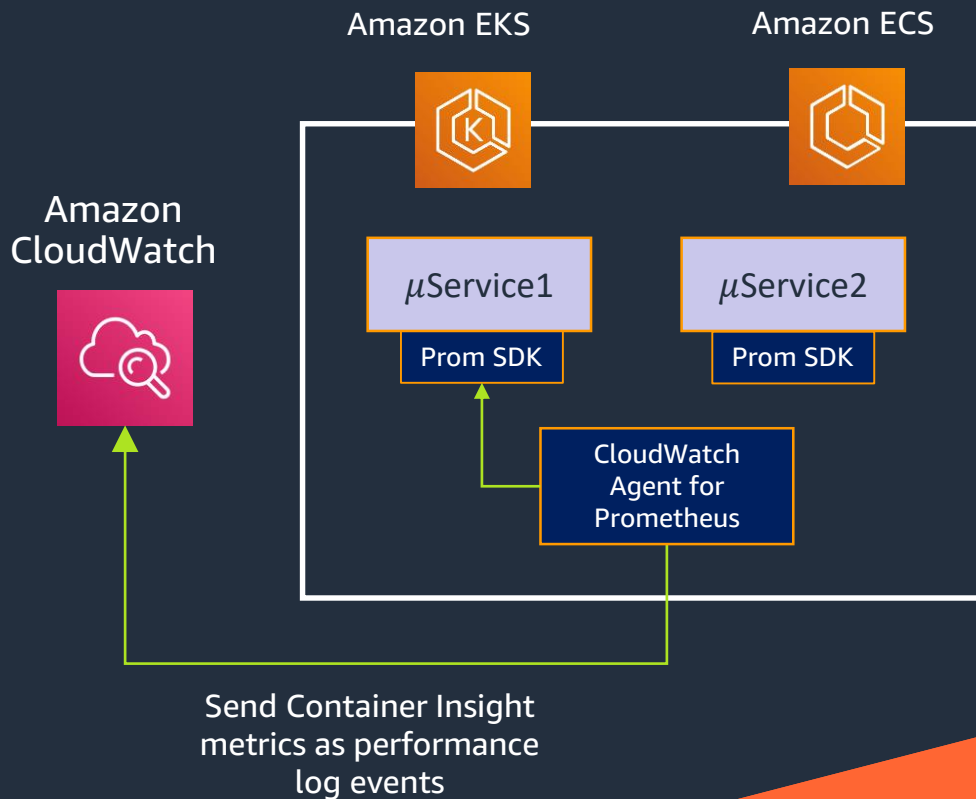
# CloudWatch Container Insights

- Collect, aggregate and summarize metrics and logs from containerized applications
- Collect instance-level metrics such as CPU, memory, disk and network usage
- Operational data collected as performance log events with EMF from which metrics are extracted



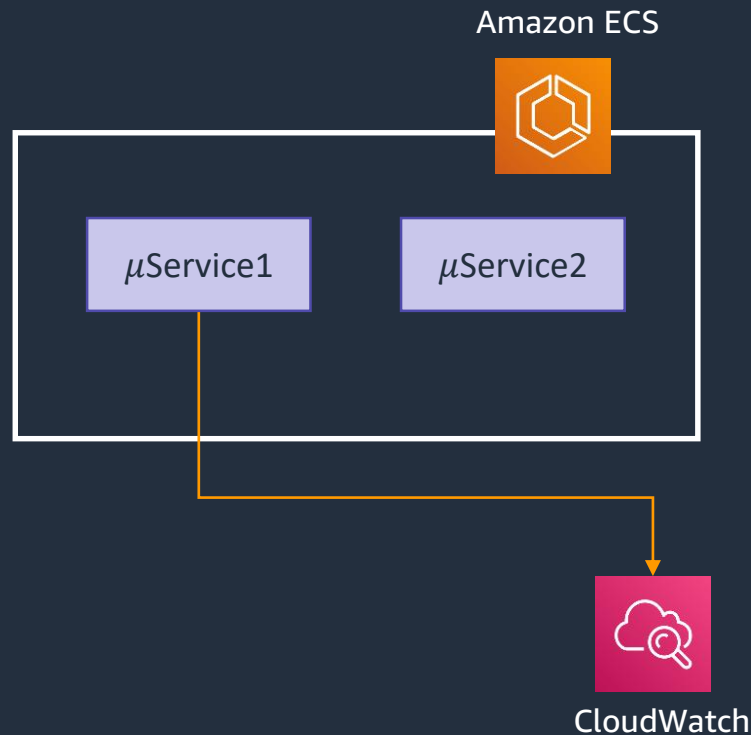
# CloudWatch Container Insights for Prometheus

- Collect, aggregate and summarize metrics and logs from containerized applications
- Collect instance-level metrics such as CPU, memory, disk and network usage
- Operational data collected as performance log events with EMF from which metrics are extracted



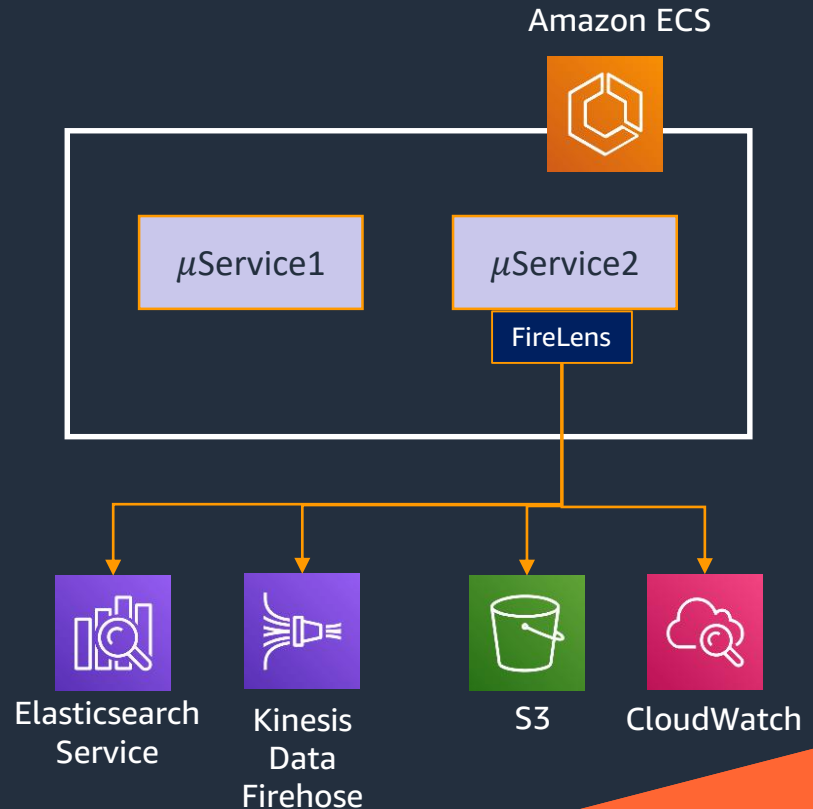
# CloudWatch logs for Amazon ECS

- Microservices running on Amazon ECS can send application logs directly to CloudWatch Logs using `awslogs` driver



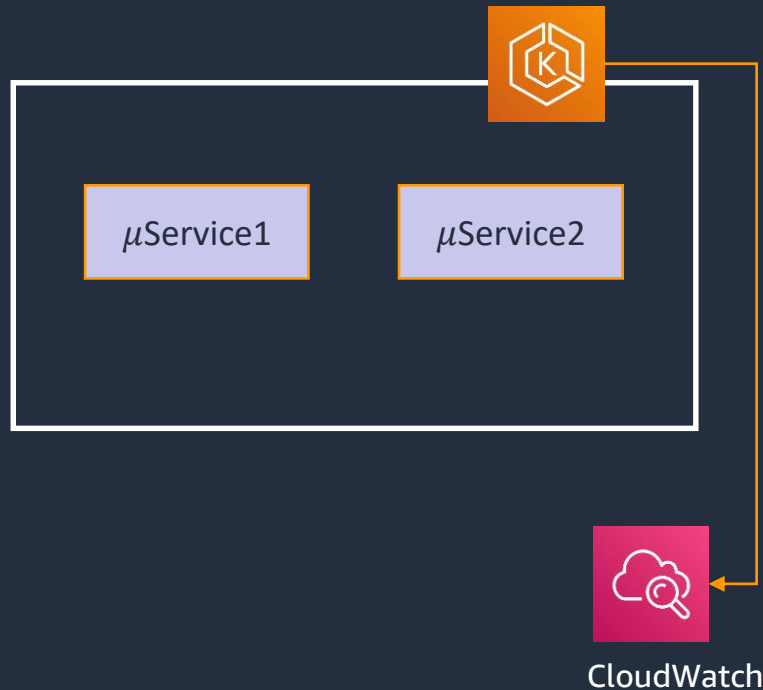
# CloudWatch logs for Amazon ECS

- Microservices running on Amazon ECS can send application logs directly to CloudWatch Logs using awslogs driver
- FireLens for ECS enables applications to send logs to many other destinations by using the awsfirelens driver; works with both FluentD and FluentBit
- Both methods work on EC2 and Fargate



# CloudWatch logs for Amazon EKS

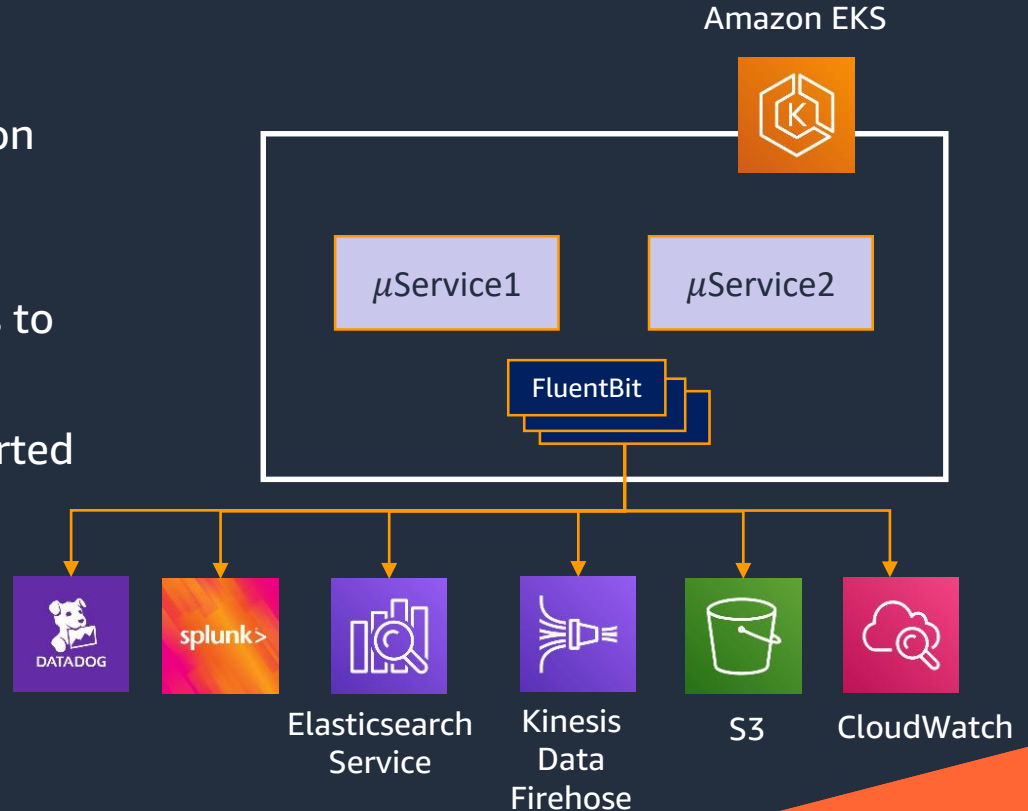
- Audit and diagnostic logs from Amazon EKS Control Plane can be sent to CloudWatch





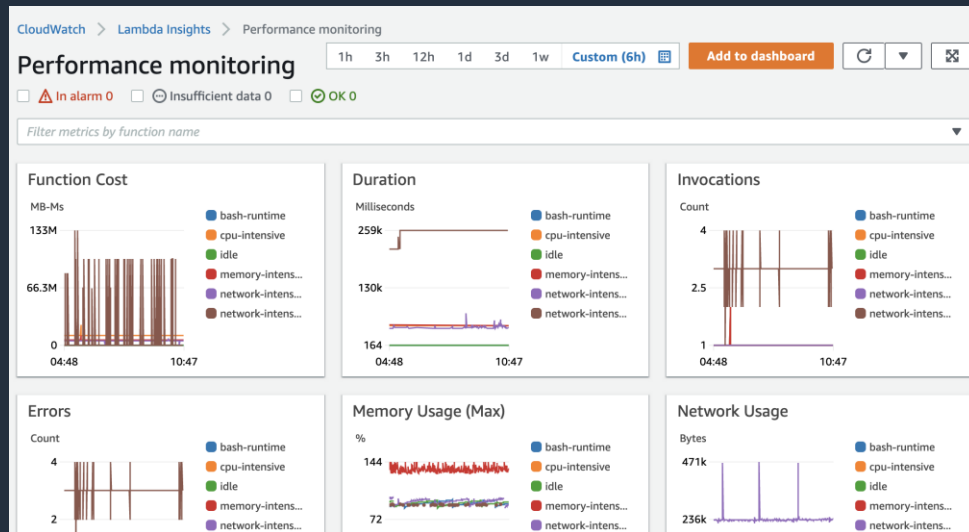
# CloudWatch logs for Amazon EKS

- Audit and diagnostic logs from Amazon EKS Control Plane can be sent to CloudWatch
- Use **FluentBit** to send application logs to destination of your choosing
- FluentBit-based logging is also supported in EKS on Fargate



# CloudWatch Lambda insights

- Get deeper insights into Lambda function executions using system-level metrics.
- Easily enabled on a per-function basis.
- Review KPIs using CloudWatch dashboard; either multi-function overview, or focus on a single function.
- Metrics are sent to CloudWatch as a single performance log event with EMF for every execution



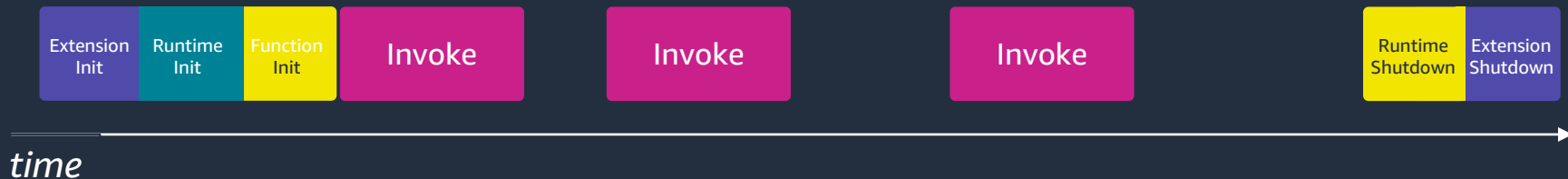
# Lambda extensions

## Receive and control Lambda lifecycle events

### Extensions

#### Primary use cases:

- Monitoring
- Configuration
- Security

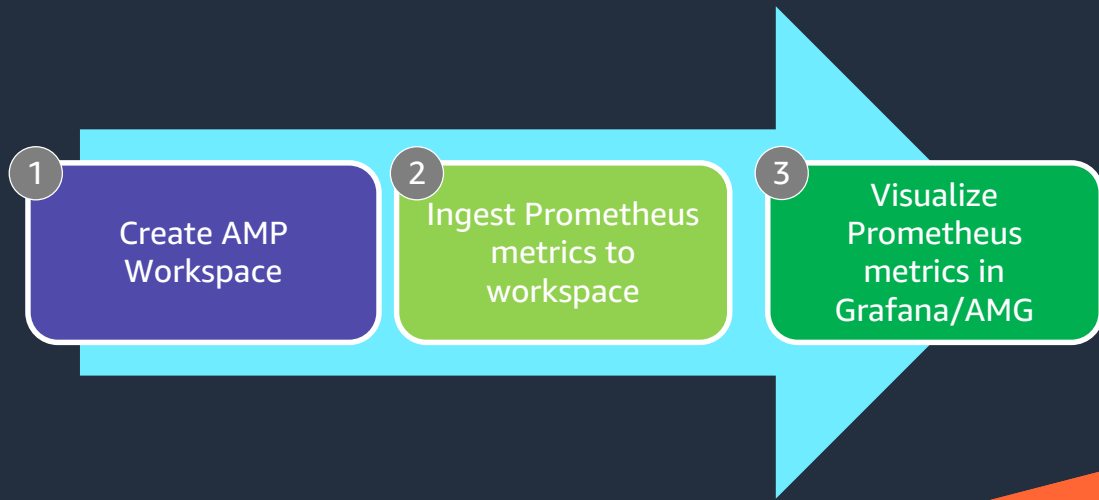


# Lambda logs API

- Send log streams to preferred destinations directly from Lambda execution environment
- Build your own
- Partner integrations:
  - Datadog
  - Lumigo
  - New Relic
  - Coralogix
  - Honeycomb
  - Sumo Logic
- Optionally disable logging to CloudWatch Logs via IAM permissions

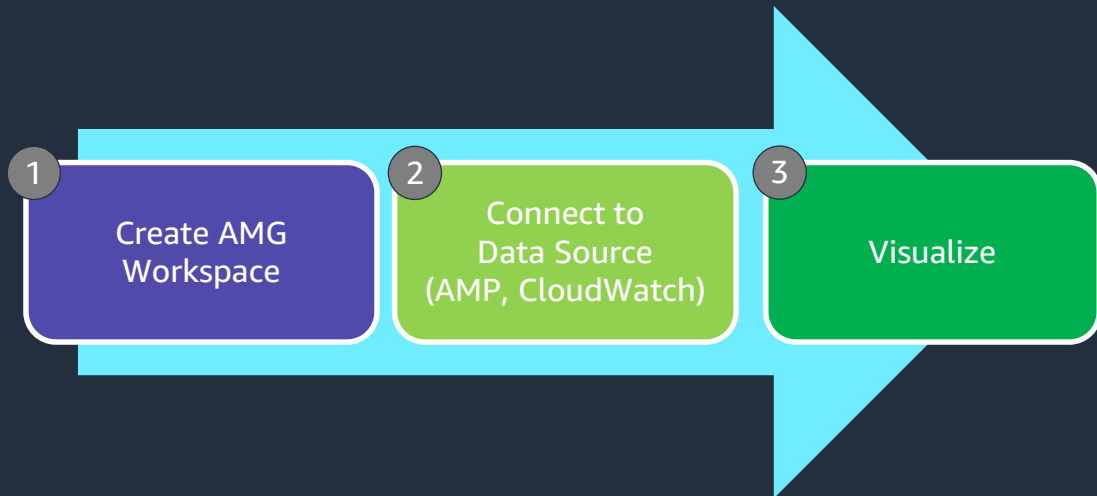
# Amazon Managed Service for Prometheus (AMP)

- Serverless Prometheus-compatible service for metrics to securely monitor container environments at scale
- Fully managed, secure, and highly available using multi-AZ deployments
- Use the same open source Prometheus data model and query language
- Improved scalability, availability, and security without managing the underlying infrastructure

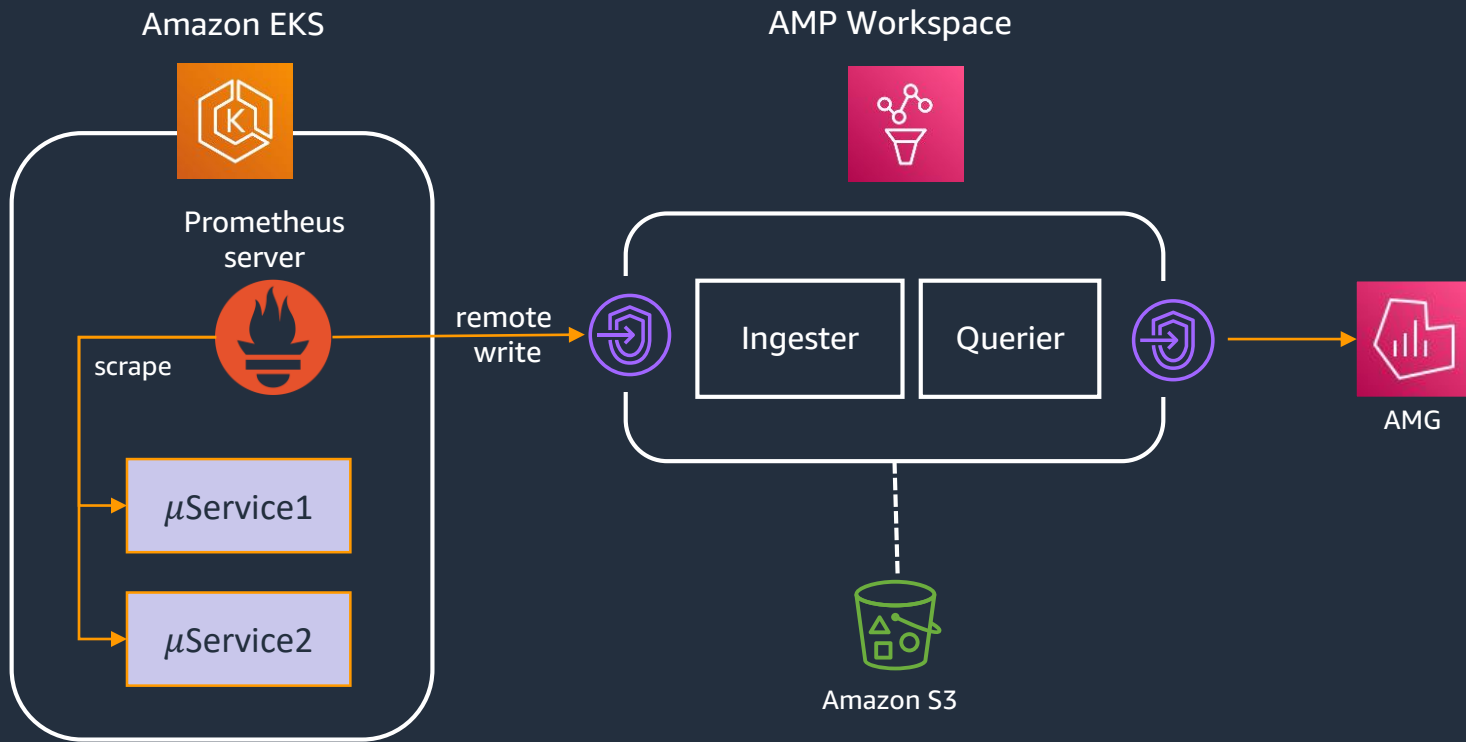


# Amazon Managed Service for Grafana (AMG)

- Scalable, secure and highly available fully-managed Grafana service
- Analyze, monitor, and alarm across multiple data sources; native AWS as well as 3<sup>rd</sup> party
- Native integration with multiple AWS Services for enterprise-ready security
- Easily upgrade to Grafana Enterprise from AWS marketplace



# Observability with AMP & AMG

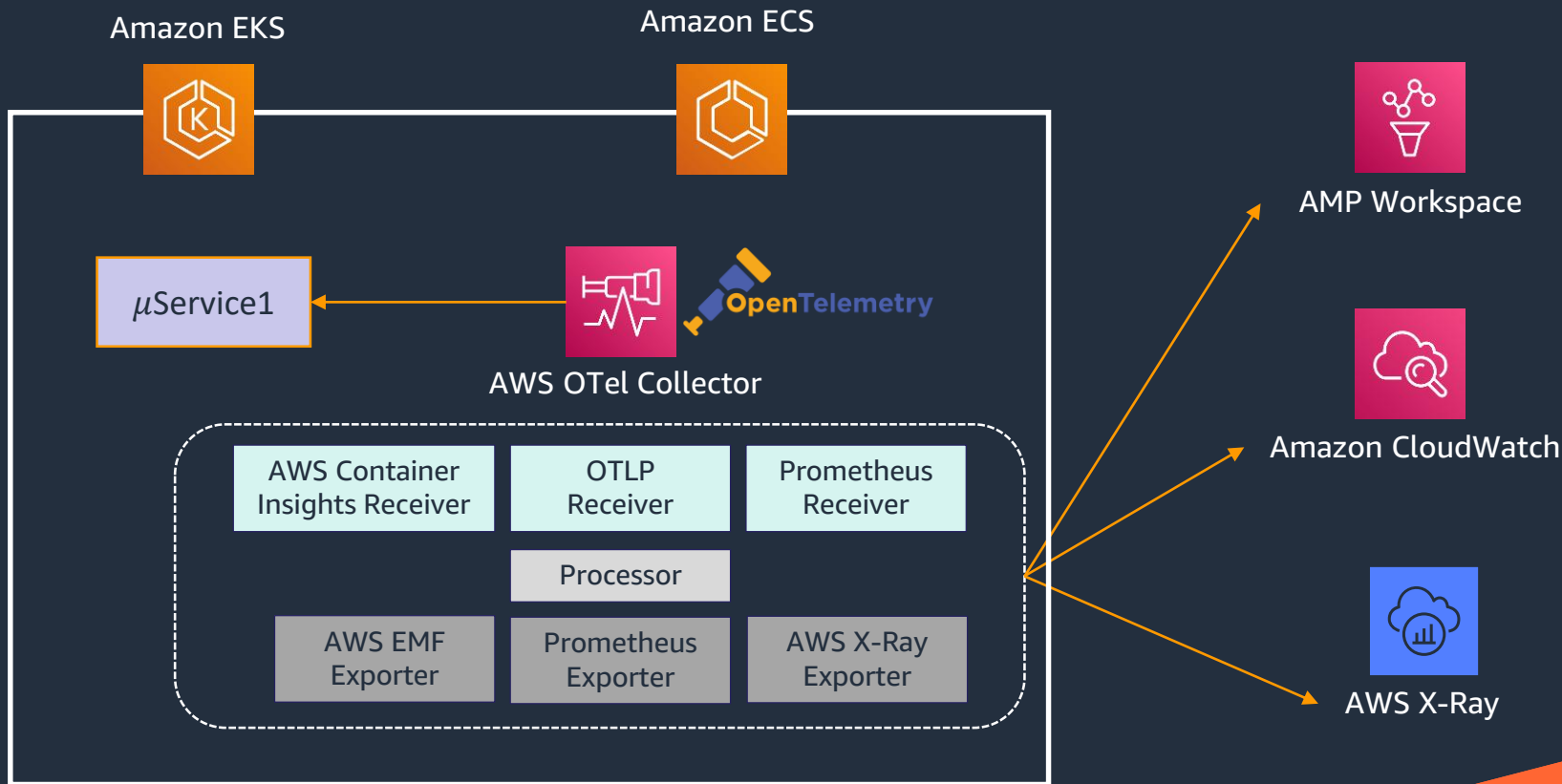


# AWS Distro for Open Telemetry (ADOT)

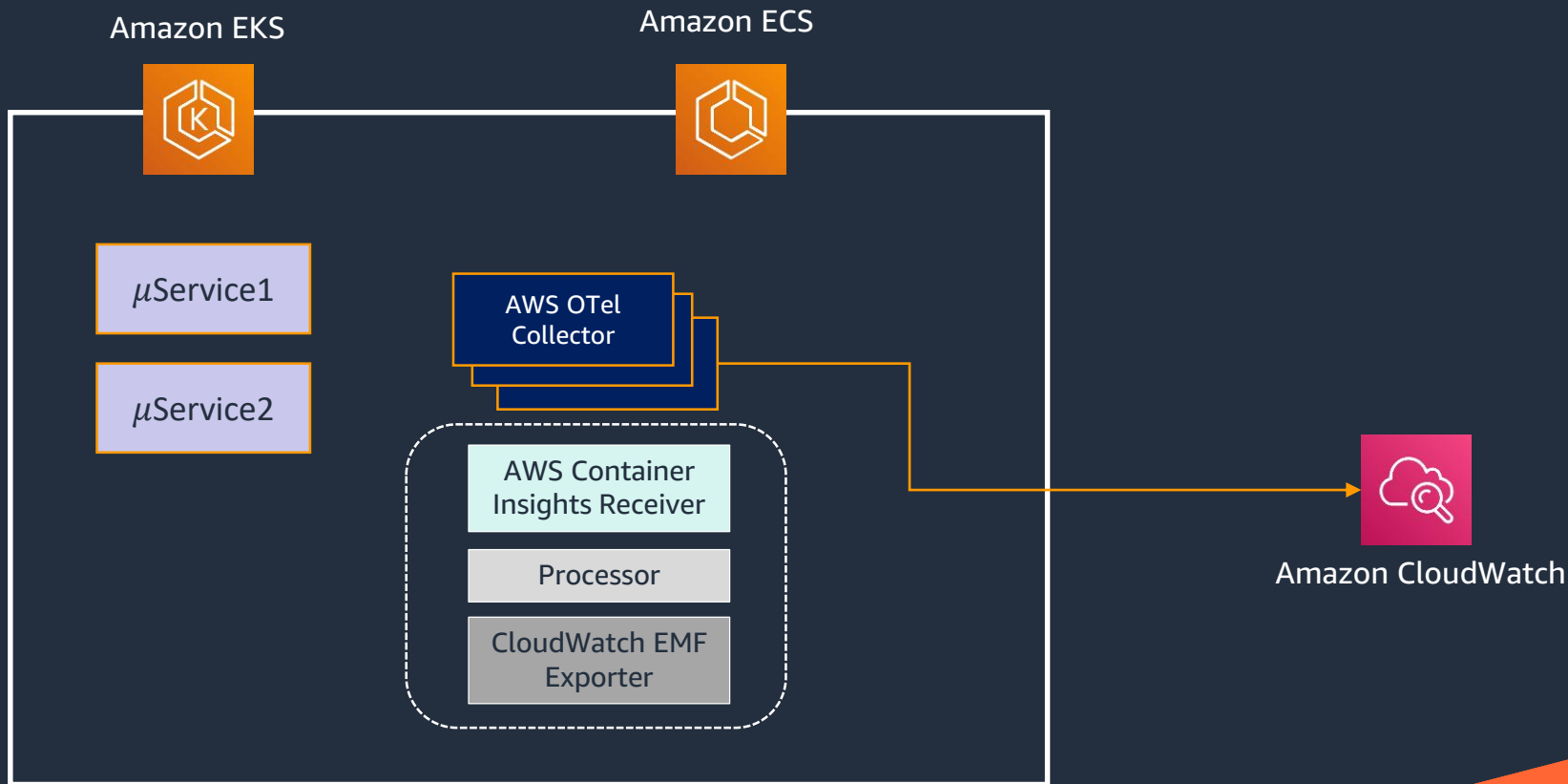
- **OpenTelemetry** provides open source APIs, libraries, and agents to collect distributed traces and metrics for application monitoring.
- **AWS Distro for OpenTelemetry**
  - Secure, production-ready AWS-supported distribution of OpenTelemetry project
  - Instrument your applications just once to send correlated metrics and traces to multiple monitoring solutions
  - Use auto-instrumentation agents to collect traces without changing your code



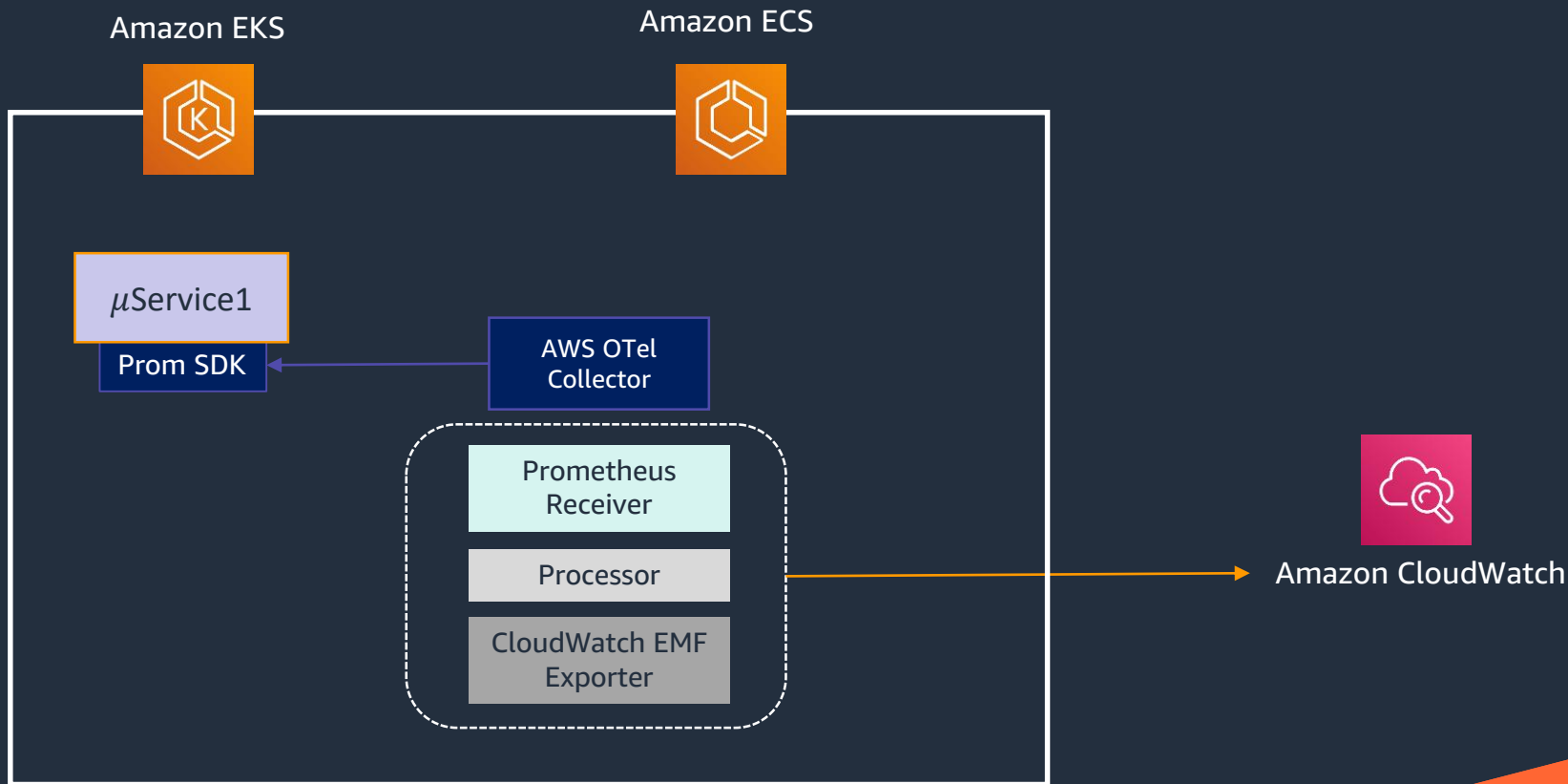
# Observability with ADOT (a.k.a OTEL)



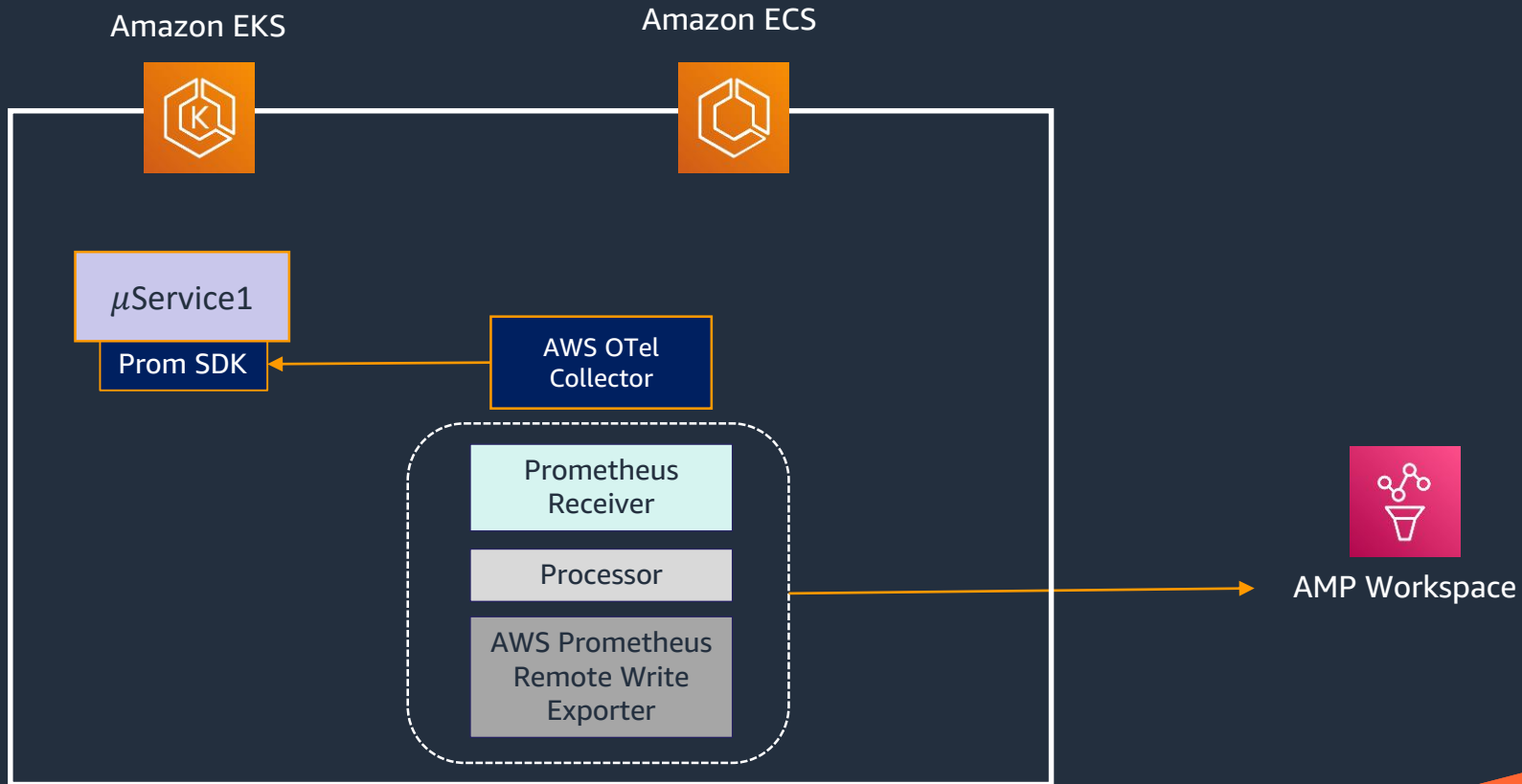
# CloudWatch Container Insights with ADOT



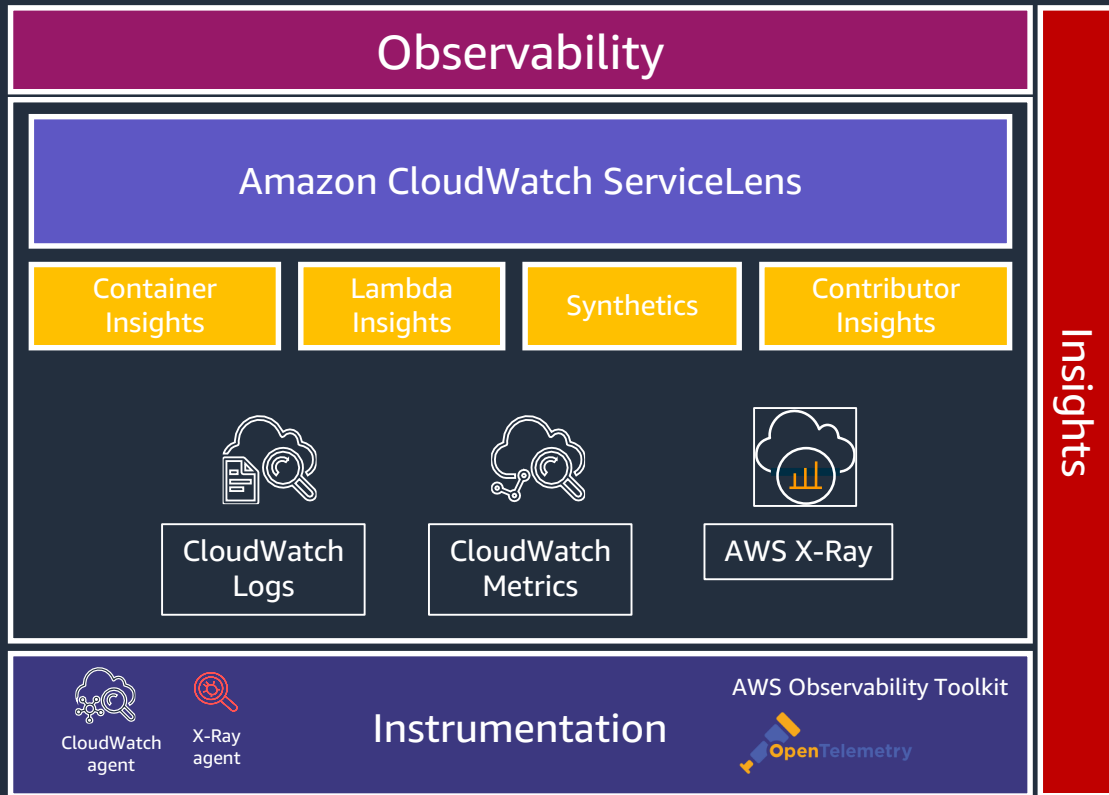
# CloudWatch Container Insights for Prometheus with ADO T



# AMP with ADOT



# Monitoring options



# Monitoring options

