
AWS
re:Invent
NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

COP344

Observability: Best practices for modern applications

Roland Barcia (he/him)

Director, WW Serverless SA
Amazon Web Services

Greg Eppel (he/him)

Tech Leader, Cloud Operations
Amazon Web Services

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

Agenda

1. Why are modern applications more difficult to observe?
2. Navigating your instrumentation options
3. Optimizing the cost of high cardinality
4. Reducing alarm fatigue
5. Avoiding dangling traces

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

Why are modern applications more difficult to observe?

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

Modern application complexity



Monolithic Application

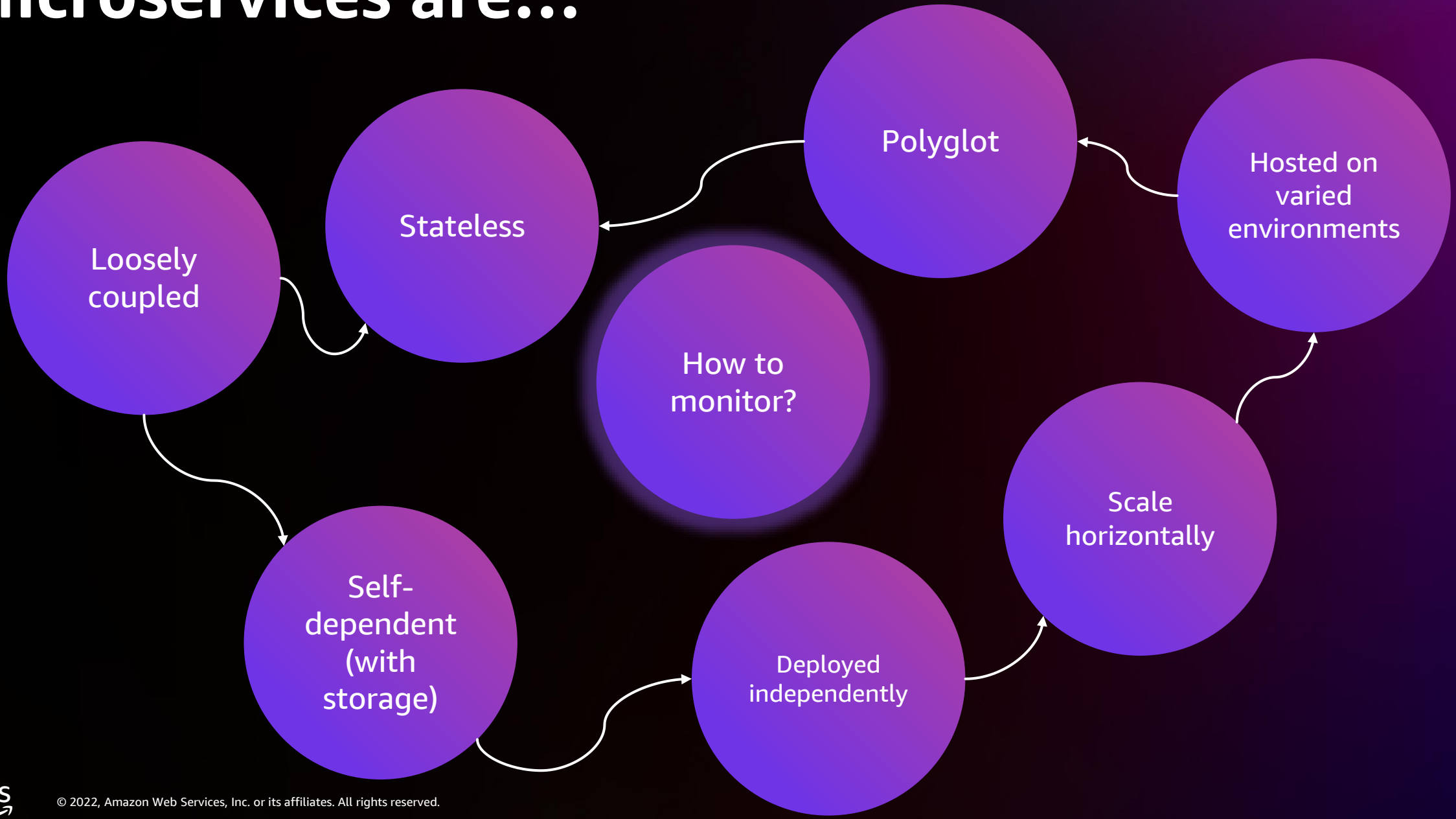
Services

Microservices

aws

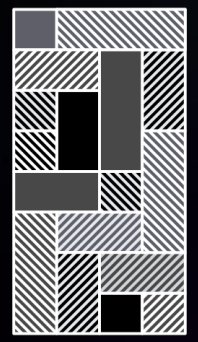
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Microservices are...



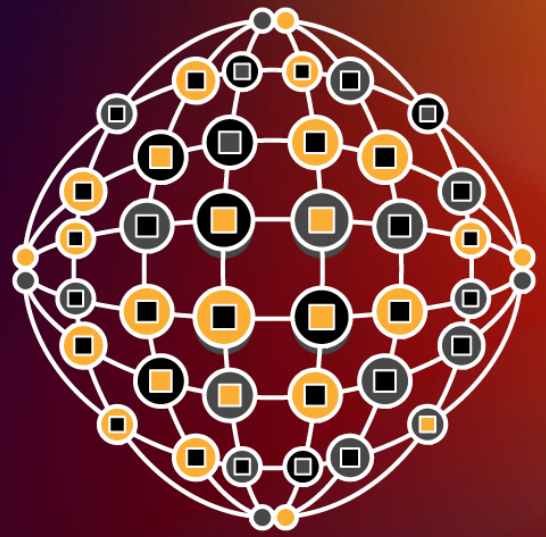
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Modern application complexity





Monolithic Application



Microservices

aws

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

Observability in software

In software systems, observability is the ability to collect, correlate, and analyze data about program execution, internal states of modules, and communication between components.

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

Observability matters because



Visibility



Real-time troubleshooting



Customer experience



Applications = $$

Operational

Business

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

“Everything fails all the time.”

Werner Vogels

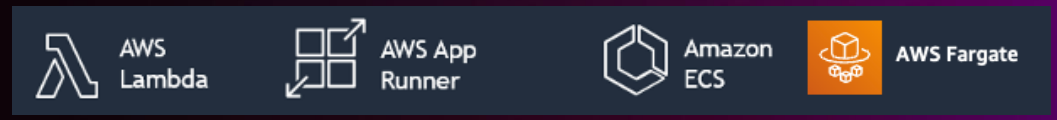
VP and CTO, Amazon

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

Strategy approaches to modern apps

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

Strategy approach



- **AWS Serverless-first approach**
  - Use AWS managed services and tools as your first choice
  - Many of our customers invest in developing discipline around AWS including provisioning, DevOps, infrastructure automation, security, networking, and observability/operations
  - Productivity and minimal operational burden
  - Includes Lambda and containers



- **Kubernetes (+++) on AWS**
  - Kubernetes as their primary compute platform interface
  - Many customers choose AWS as their primary place to run Kubernetes
  - Adopt discipline around running and managing several Kubernetes clusters and the workload and tools on them – advanced patterns like GitOps
  - Rich ecosystem and partner tools
  - Many Kubernetes customers make use of managed AWS services and other compute options like lambda for certain use cases

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

AWS Observability options

Observability

AWS native services

Amazon CloudWatch ServiceLens

Container insights | Lambda insights | Contributor insights | Application insights

Synthetics | Dashboards | Alarms | RUM

Metrics | Logs | AWS X-Ray

Open source managed services

Amazon Managed Grafana

Do it yourself (DIY) – AWS OSS solutions

Amazon OpenSearch Service | Amazon Managed Service for Prometheus | Jaeger and Zipkin Tracing

Insights & ML

CloudWatch agent | AWS X-Ray agent | Collectors and SDKs | AWS Distro for OpenTelemetry

Instrumentation

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

Best practices



Navigating your instrumentation options



Optimizing the cost of high cardinality



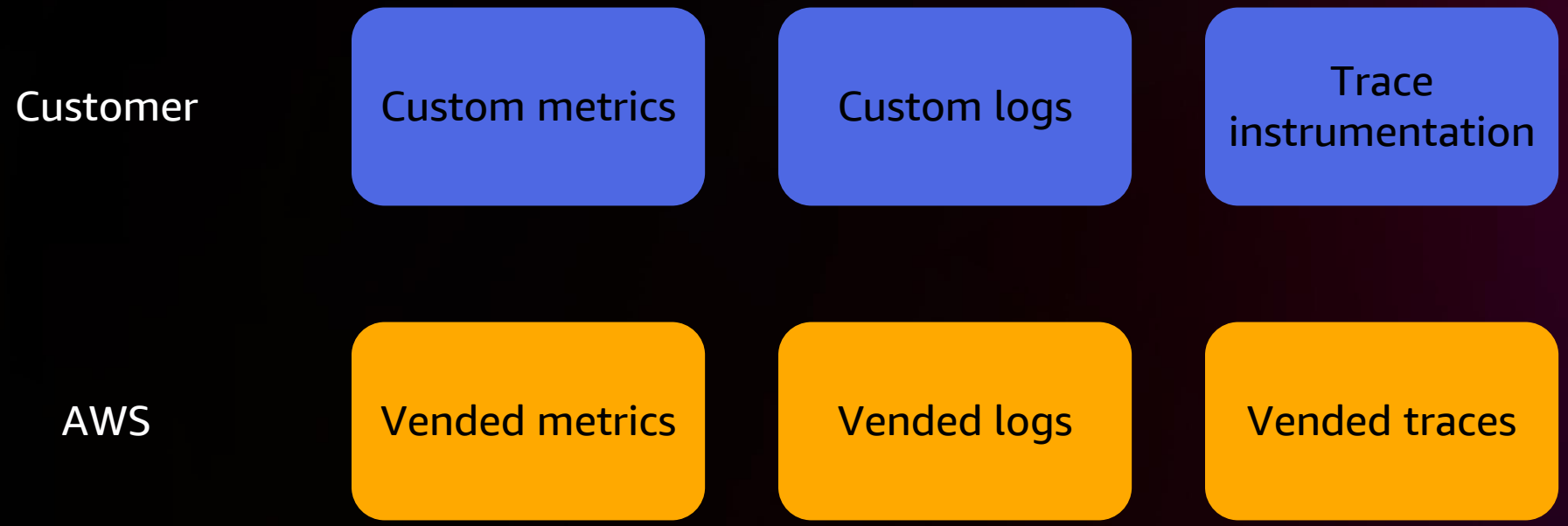
Reducing alarm fatigue



Avoiding dangling traces

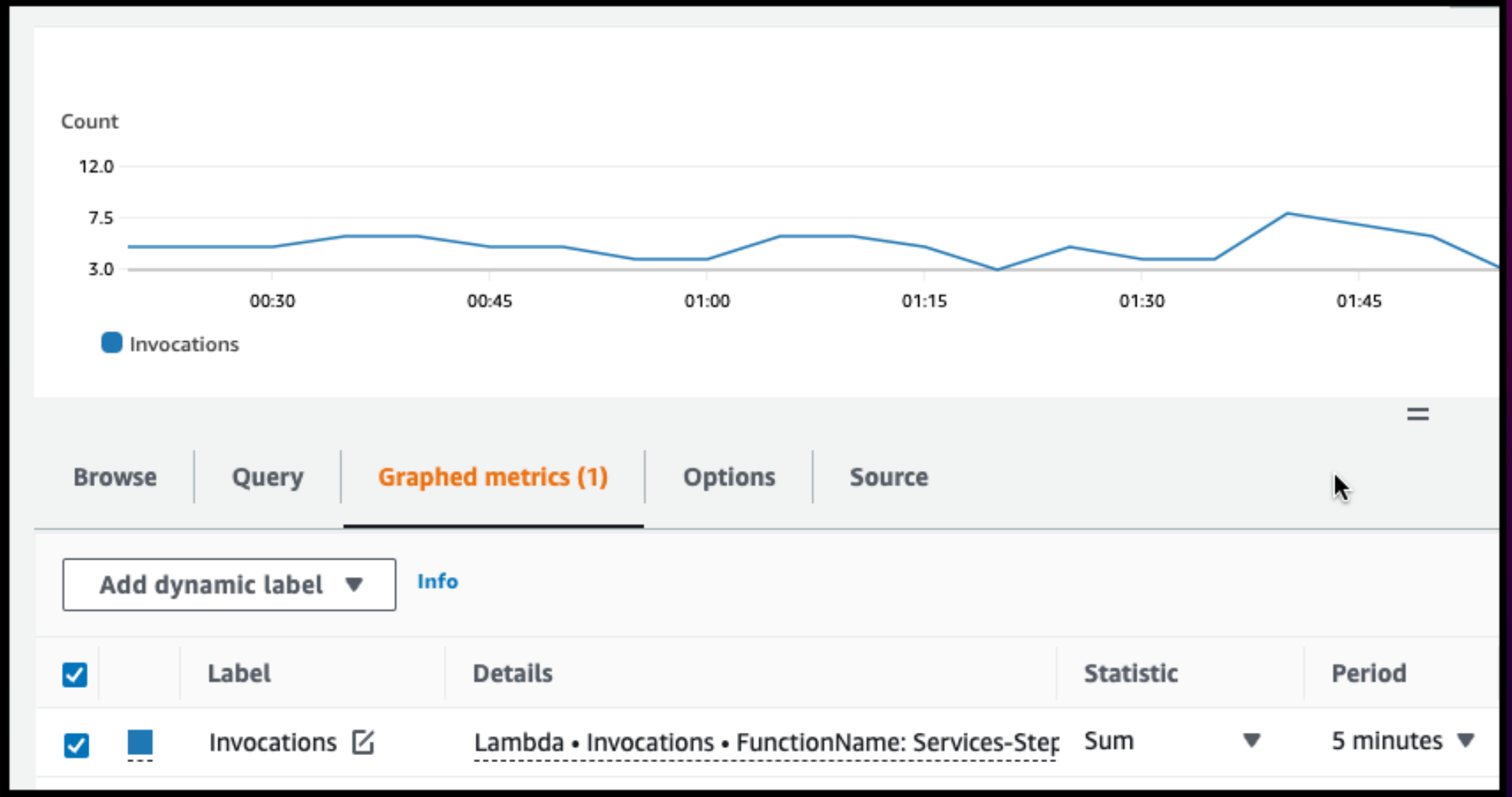
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Instrumentation is a shared responsibility



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

Vended metrics



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

Vended metrics

AWS services that publish CloudWatch metrics

PDF | RSS

The following AWS services publish metrics to CloudWatch. For information about the metrics and dimensions, see the specified documentation.

Service	Namespace	Documentation
Amazon Kinesis Data Firehose	AWS/Firehose	Monitoring Kinesis Data Firehose Using CloudWatch Metrics
Amazon Kinesis Data Streams	AWS/Kinesis	Monitoring Amazon Kinesis Data Streams with Amazon CloudWatch
Amazon Kinesis Video Streams	AWS/KinesisVideo	Monitoring Kinesis Video Streams Metrics with CloudWatch
AWS Lambda	AWS/Lambda	AWS Lambda Metrics
Amazon Lex	AWS/Lex	Monitoring Amazon Lex with Amazon CloudWatch
Amazon Location Service	AWS/Location	Amazon Location Service metrics exported to Amazon CloudWatch
Amazon Lookout for Metrics	AWS/LookoutMetrics	Monitoring Lookout for Metrics with Amazon CloudWatch
Amazon Machine Learning	AWS/ML	Monitoring Amazon ML with CloudWatch Metrics
Amazon Managed Streaming for Apache Kafka	AWS/Kafka	Monitoring Amazon MSK with Amazon CloudWatch
Amazon MQ	AWS/AmazonMQ	Monitoring Amazon MQ Brokers Using Amazon CloudWatch
Amazon Neptune	AWS/Neptune	Monitoring Neptune with CloudWatch
AWS Network Firewall	AWS/NetworkFirewall	AWS Network Firewall metrics in Amazon CloudWatch
AWS Network Manager	AWS/NetworkManager	CloudWatch metrics for on-premises resources
Amazon Nimble Studio	AWS/NimbleStudio	Monitoring Nimble Studio with Amazon CloudWatch
AWS OpsWorks	AWS/OpsWorks	Monitoring Stacks using Amazon CloudWatch
Amazon Polly	AWS/Polly	Integrating CloudWatch with Amazon Polly
AWS PrivateLink	AWS/PrivateLinkEndpoints	CloudWatch metrics for AWS PrivateLink

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

Vended metrics

Using invocation metrics

Invocation metrics are binary indicators of the outcome of an invocation. For example, if the function returns an error, Lambda sends the `Errors` metric with a value of 1. To get a count of the number of function errors that occurred each minute, view the `Sum` of the `Errors` metric with a period of 1 minute.

View the following metrics with the `Sum` statistic.

Invocation metrics

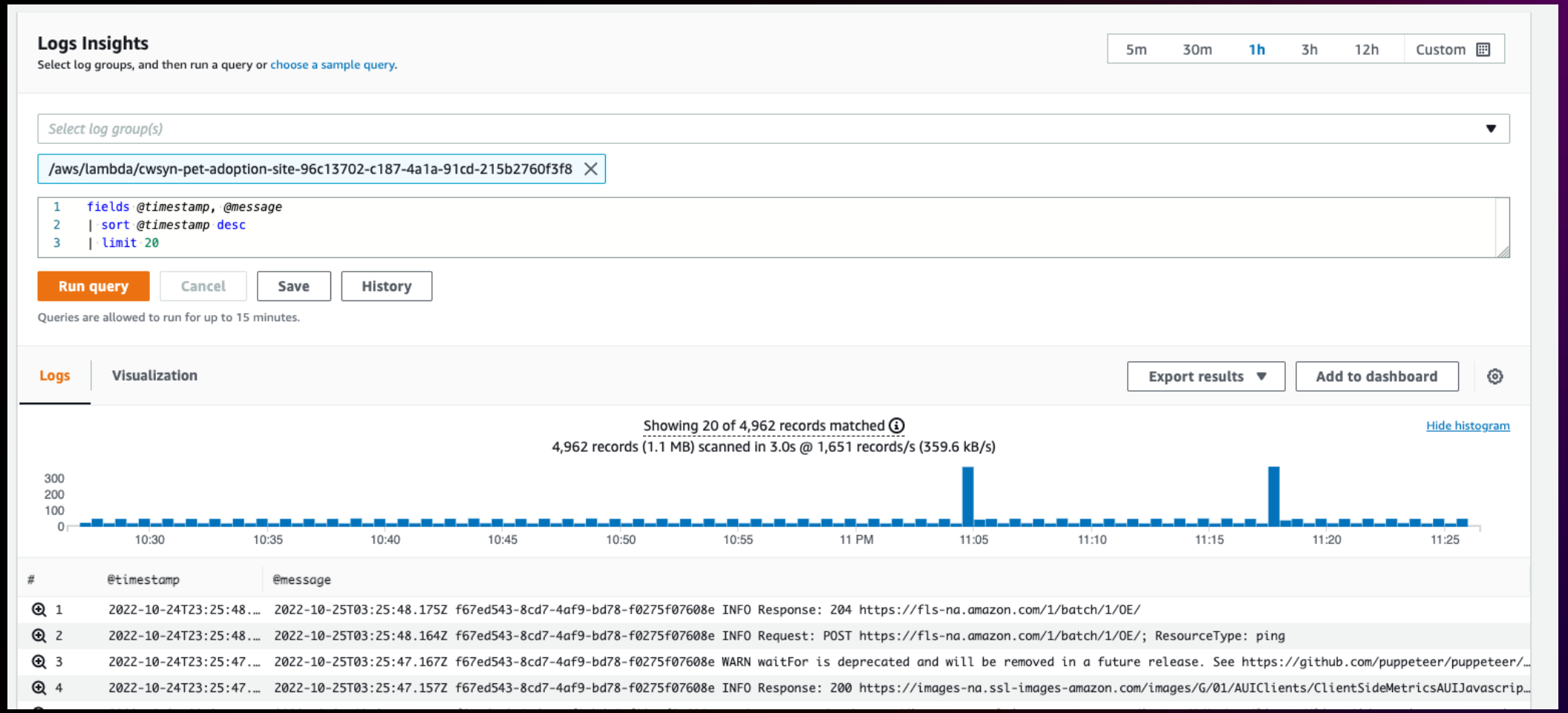
- `Invocations` – The number of times that your function code is invoked, including successful invocations and invocations that result in a function error. Invocations aren't recorded if the invocation request is throttled or otherwise results in an invocation error. This equals the number of requests billed.
- `Errors` – The number of invocations that result in a function error. Function errors include exceptions that your code throws and exceptions that the Lambda runtime throws. The runtime returns errors for issues such as timeouts and configuration errors. To calculate the error rate, divide the value of `Errors` by the value of `Invocations`. Note that the timestamp on an error metric reflects when the function was invoked, not when the error occurred.

Choose a dimension.

- **By Function Name** (`FunctionName`) – View aggregate metrics for all versions and aliases of a function.
- **By Resource** (`Resource`) – View metrics for a version or alias of a function.
- **By Executed Version** (`ExecutedVersion`) – View metrics for a combination of alias and version. Use the `ExecutedVersion` dimension to compare error rates for two versions of a function that are both targets of a weighted alias.
- **Across All Functions** (none) – View aggregate metrics for all functions in the current AWS Region.

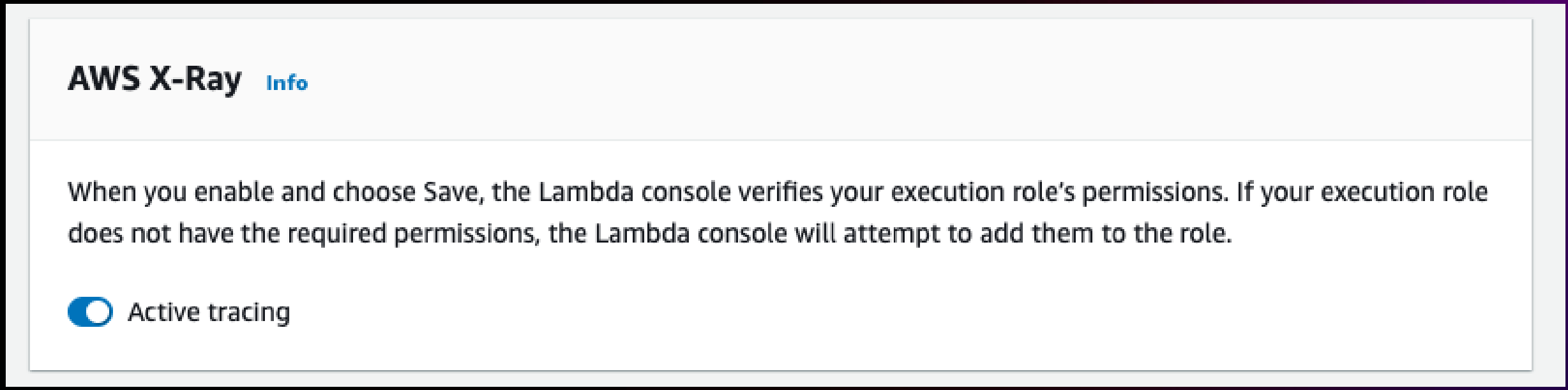
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Vended logs



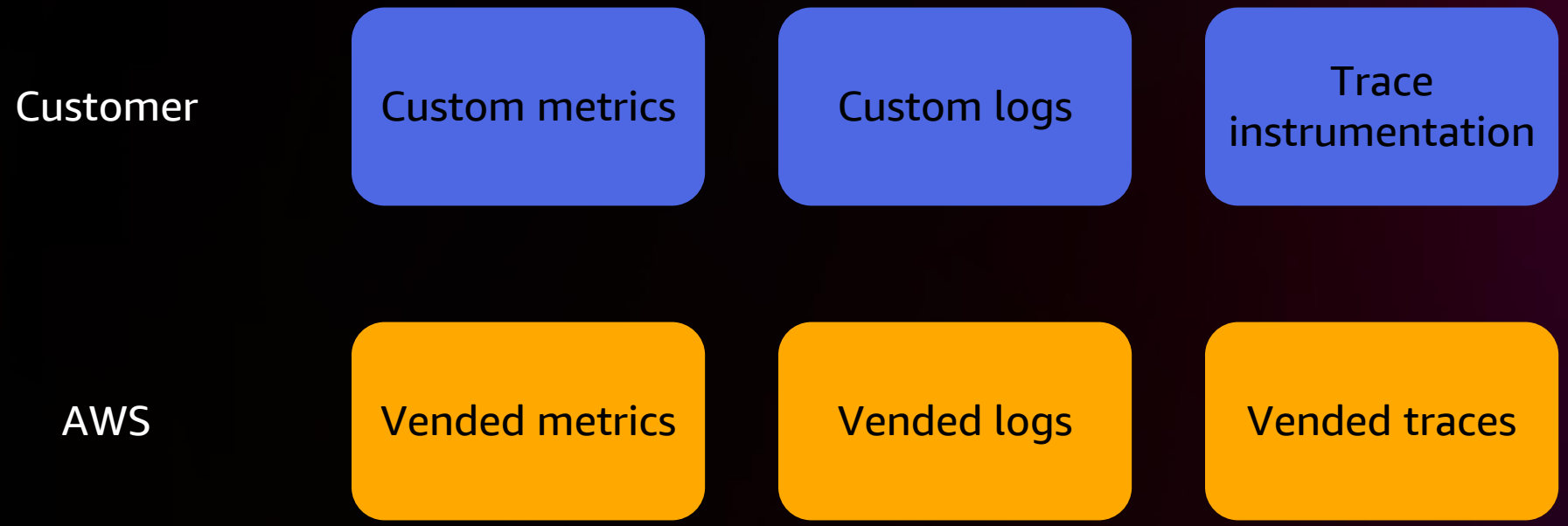
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Vended traces



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

Instrumentation is a shared responsibility



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

Custom instrumentation



AWS Lambda



Amazon Elastic Container Service (Amazon ECS)

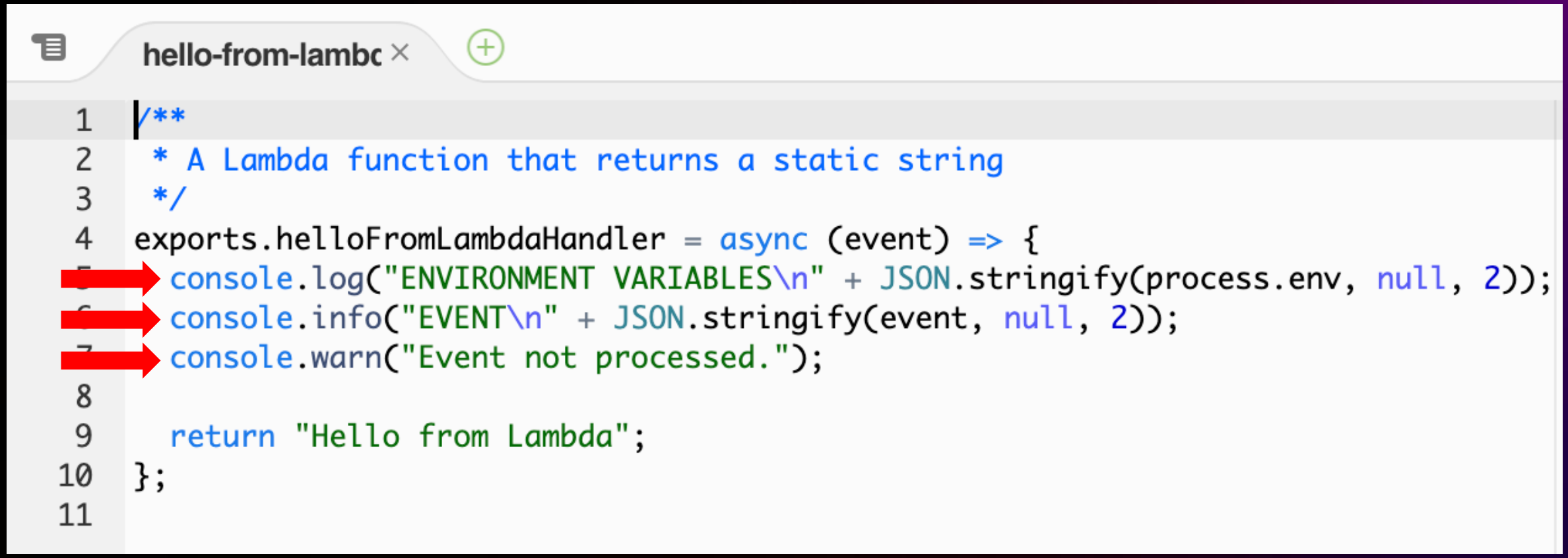


Amazon Elastic Kubernetes Service (Amazon EKS)

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

Lambda instrumentation

LOGS



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

Lambda instrumentation

METRICS

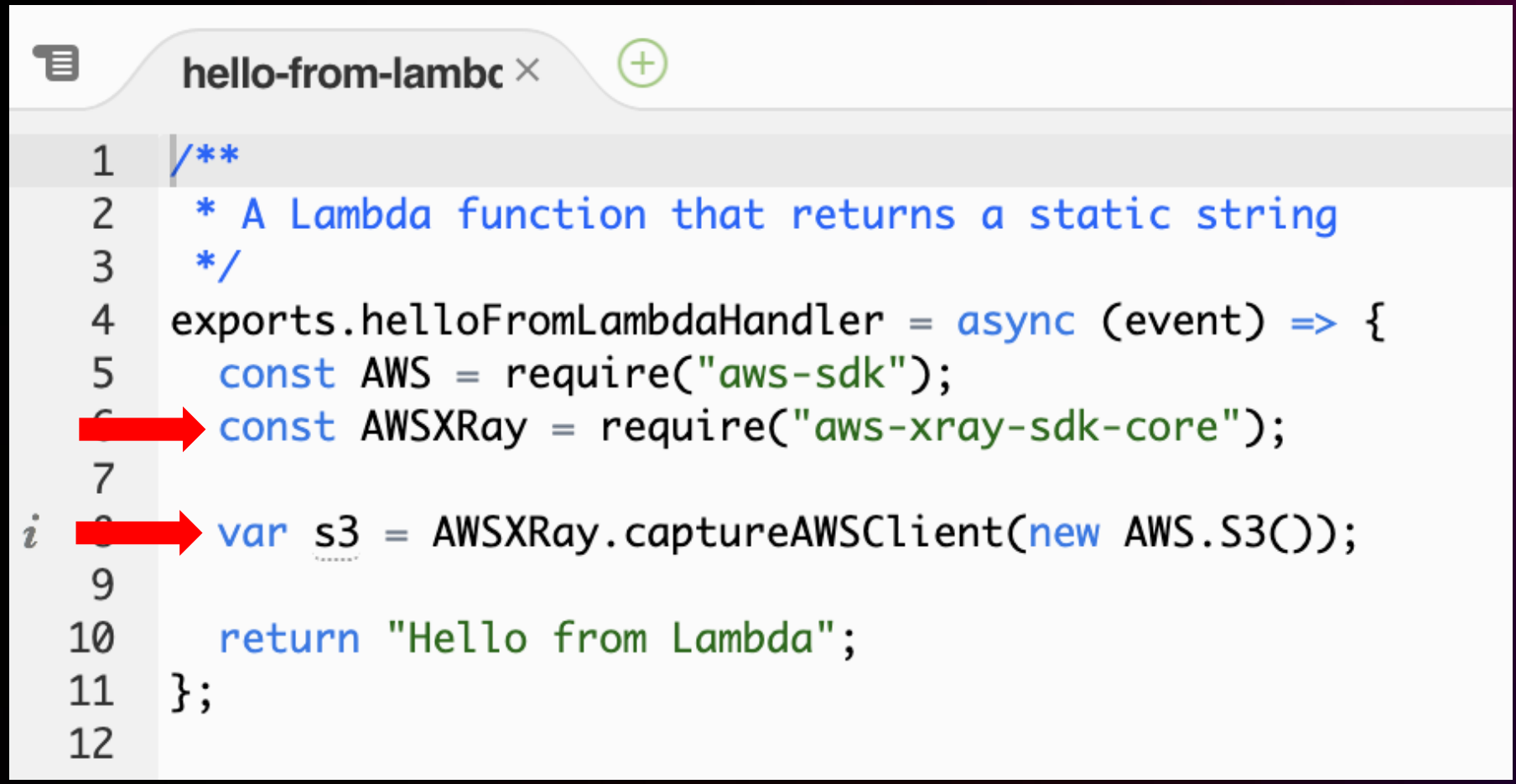
hello-from-lambc

```
1  /**
2   * A Lambda function that returns a static string
3   */
4  exports.helloFromLambdaHandler = async () => {
5    const { metricScope, Unit } = require("aws-embedded-metrics");
6
7    const myFunc = metricScope((metrics) => async () => {
8      metrics.putDimensions({ Service: "Aggregator" });
9      metrics.putMetric("ProcessingLatency", 100, Unit.Milliseconds);
10     metrics.setProperty("RequestId", "422b1569-16f6-4a03-b8f0-fe3fd9b100f8");
11   });
12
13   await myFunc();
14
15   return "Hello from Lambda";
16 };
```

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

Lambda instrumentation

TRACES



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

Instrumentation



AWS Lambda



Amazon Elastic Container Service (Amazon ECS)



Amazon Elastic Kubernetes Service (Amazon EKS)

aws

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

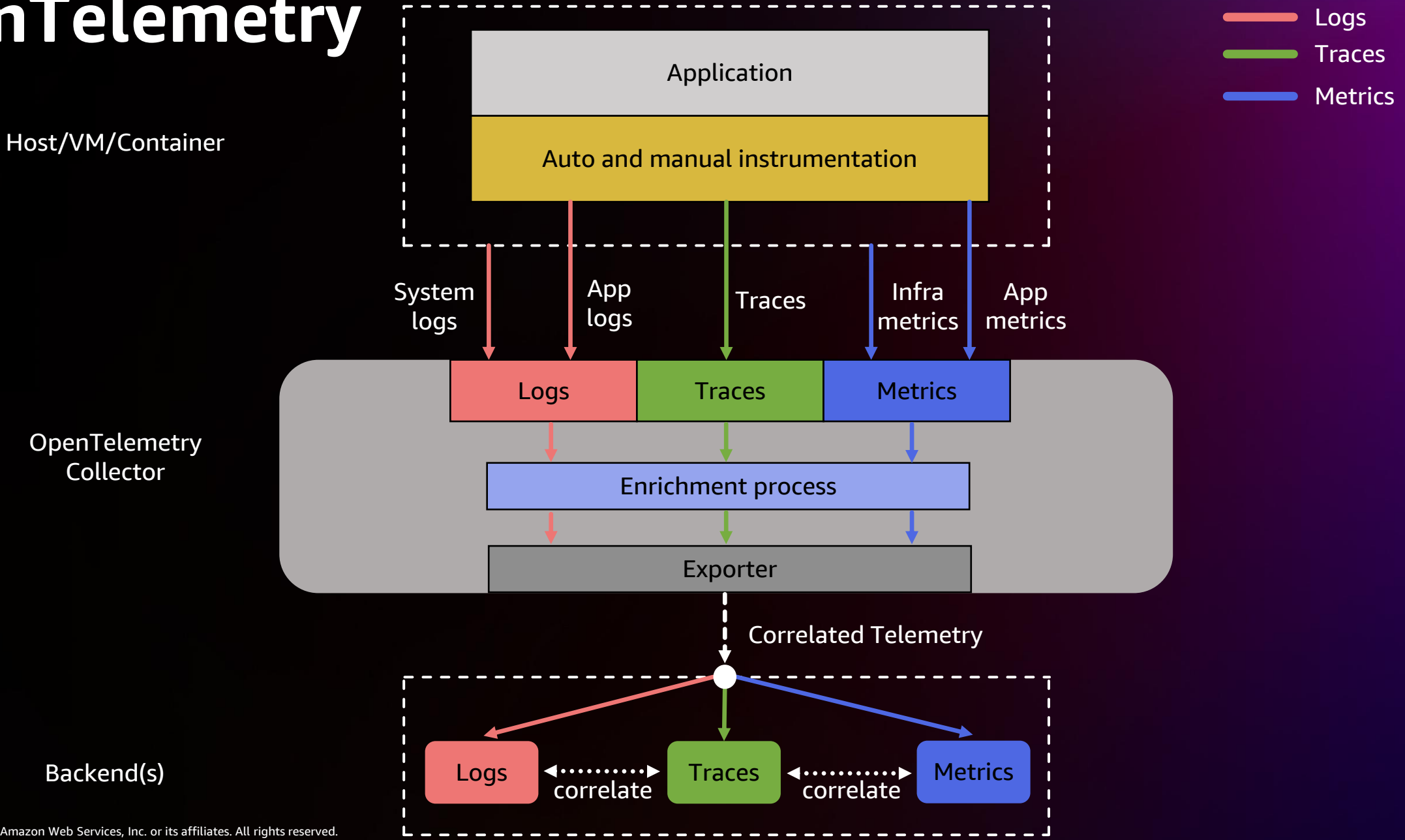
Instrumentation agents

	Logs	Metrics	Traces
AWS Distro for OpenTelemetry	✗	✓	✓
CloudWatch agent	✓	✓	✗
X-Ray Daemon	✗	✗	✓
FireLens	✓	✗	✗
Fluent Bit	✓	✗	✗

aws

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

OpenTelemetry



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

Instrumentation takeaway



SUMMARY

1. Use AWS Distro for OpenTelemetry (ADOT) for metrics and traces

2. Choose a logging agent (CloudWatch or FireLens agent)

3. OpenTelemetry Specification

   Traces: GA

   Metrics: GA

   Logs: Draft

4. In the future, use ADOT for logs as well

aws

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

Best practices



Navigating your instrumentation options



Optimizing the cost of high cardinality



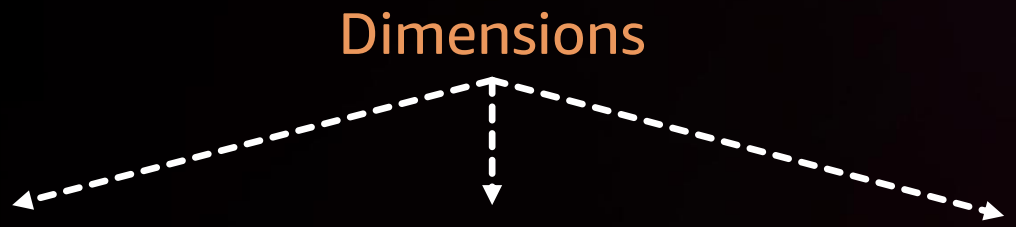
Reducing alarm fatigue

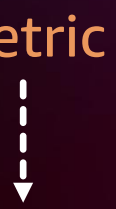


Avoiding dangling traces

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

High cardinality telemetry



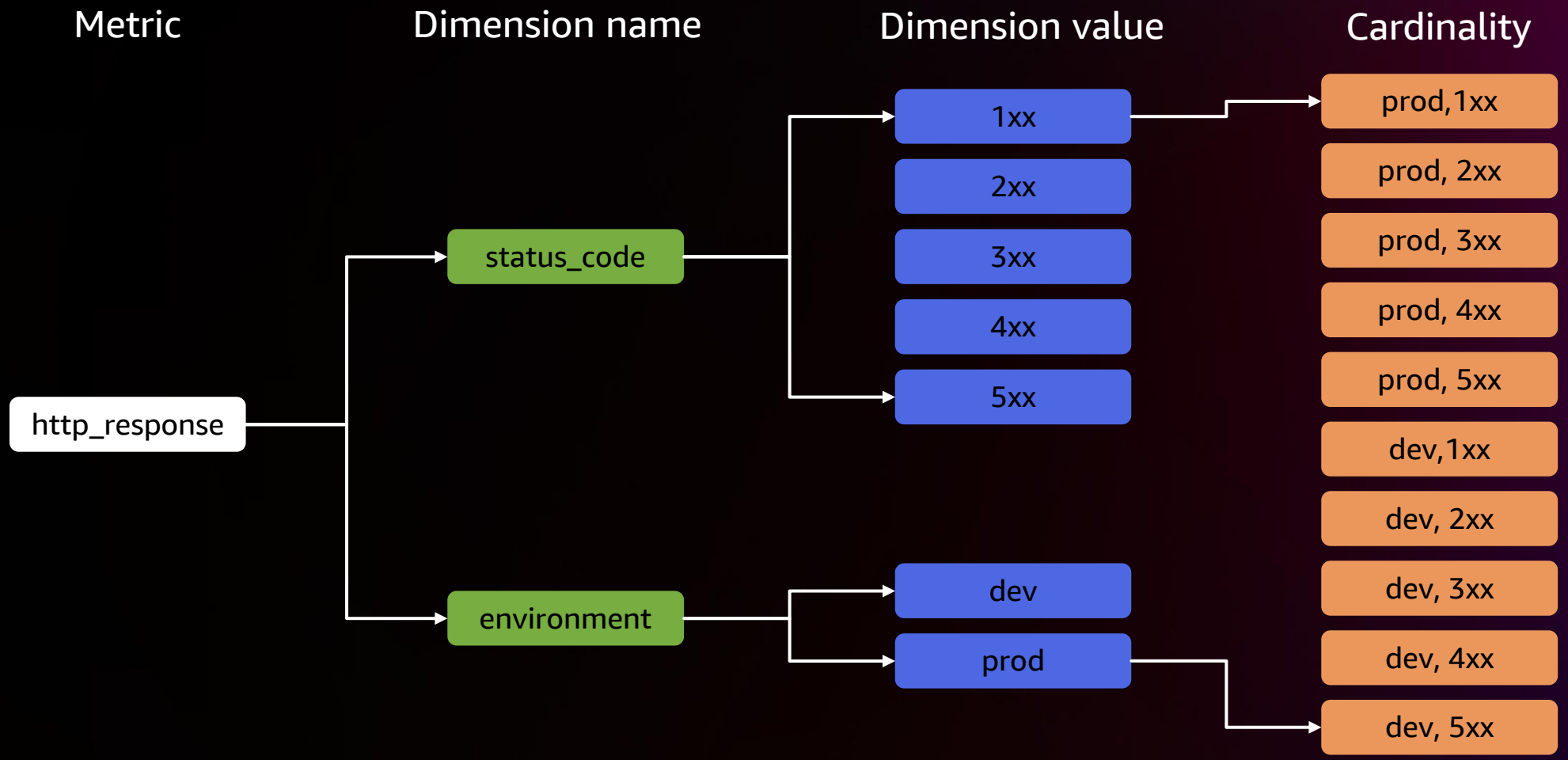


RequestID	CustomerID	Service	Latency (ms)
63VA3X6D11SJ	434124226	PutItem	152
2HQ0NZPSA20W	616095267	ListItems	422
96M1IMKSU4JK	500142823	GetItem	384
RYUEP2GTPI4A	379212302	ListItems	883
BAD3Y3OXW7AC	677124695	GetItem	231



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

High cardinality growth



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

High cardinality telemetry



http_response

- 2 environments
- 10 services
- 5 status codes
- 10 instances

VM Environment

1,000 possible metrics

$300/month



http_response

- 2 environments
- 100 services
- 5 status codes
- 10,000 tasks/pods

Cloud Native Environment

10,000,000 possible metrics

$244,500/month

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

High cardinality telemetry

METRICS OR LOGS

Metrics



- Fast to query
- Drives alarms

and

or

Logs



- Cost-efficient
- No retention period

with

CloudWatch Embedded Metric Format

aws

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

High cardinality telemetry

BEST OF BOTH WORLDS WITH LOGS AND METRICS



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

CloudWatch Embedded Metric Format demo

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

High cardinality telemetry



SUMMARY

1. Identify potential high cardinality dimensions
2. Ingest your telemetry as a log first
3. Then create metrics using appropriate dimensions
4. If you are using CloudWatch, leverage Embedded Metric Format
5. OpenTelemetry has an Embedded Metric Format exporter

aws

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

Best practices



Navigating your instrumentation options



Optimizing the cost of high cardinality



Reducing alarm fatigue

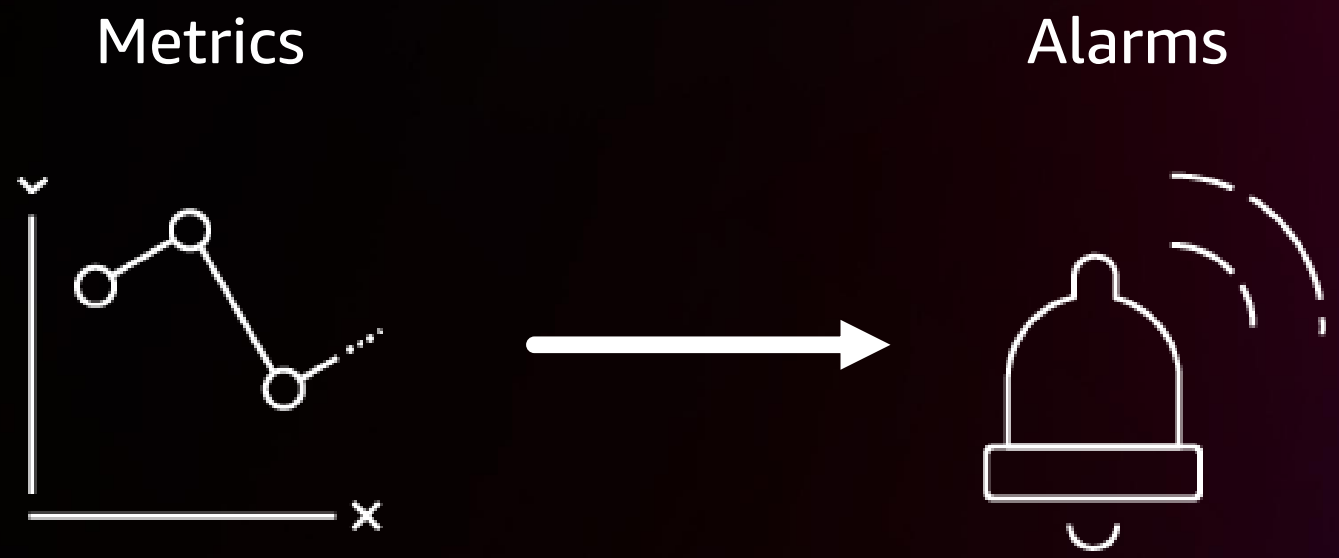


Avoiding dangling traces

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

Reducing alarm fatigue

METRICS OR LOGS



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

High cardinality telemetry



http_response

- 2 environments
- 10 services
- 5 status codes
- 10 instances

VM Environment



http_response

- 2 environments
- 100 services
- 5 status code
- 10,000 tasks/pods

Cloud Native Environment

1,000 possible metrics

10,000,000 possible metrics

aws

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



Reducing alarm fatigue

BEST PRACTICES

1. Alarm on key performance indicators (KPIs) and workload metrics
   - Order rate
   - Orders processed
   - Queue depth
2. Alert when workload outcomes are at risk
3. Alert when workload anomalies are detected
4. Automate responses to events

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

Correlate your alarms

BEST PRACTICES

1. High CPU is not a great alarm
2. Memory utilization is not a great alarm
3. However, if both are happening at the same time, that may be a good alarm
4. On a Microsoft IIS web server, this means an application pool is recycling
5. This pauses HTTP request processing, and users are going to notice

aws

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

Alarm on anomalies

BEST PRACTICES

1. Leverage statistical and machine learning algorithms

2. Generate an anomaly detection model to show an expected value range

3. Alarm on a dynamic threshold instead of a static threshold

4. There may be valid reasons for spikes and dips in your metrics; you don’t want false positives

aws

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

CloudWatch Composite Alarms and CloudWatch Anomaly Detection demo

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

Reducing alarm fatigue

SUMMARY



1. Alarm on KPI and workload metrics

2. Alarm when your workload outcomes are at risk
   - CloudWatch Synthetic Canaries

3. Correlate your alarms and notify a person when that correlation happens
   - CloudWatch Composite Alarms

4. Leverage machine learning algorithms to detect anomalies
   - CloudWatch Anomaly Detection

aws

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

Best practices



Navigating your instrumentation options



Optimizing the cost of high cardinality



Reducing alarm fatigue



Avoiding dangling traces

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

Trace propagation

WHAT IS A TRACE ID?

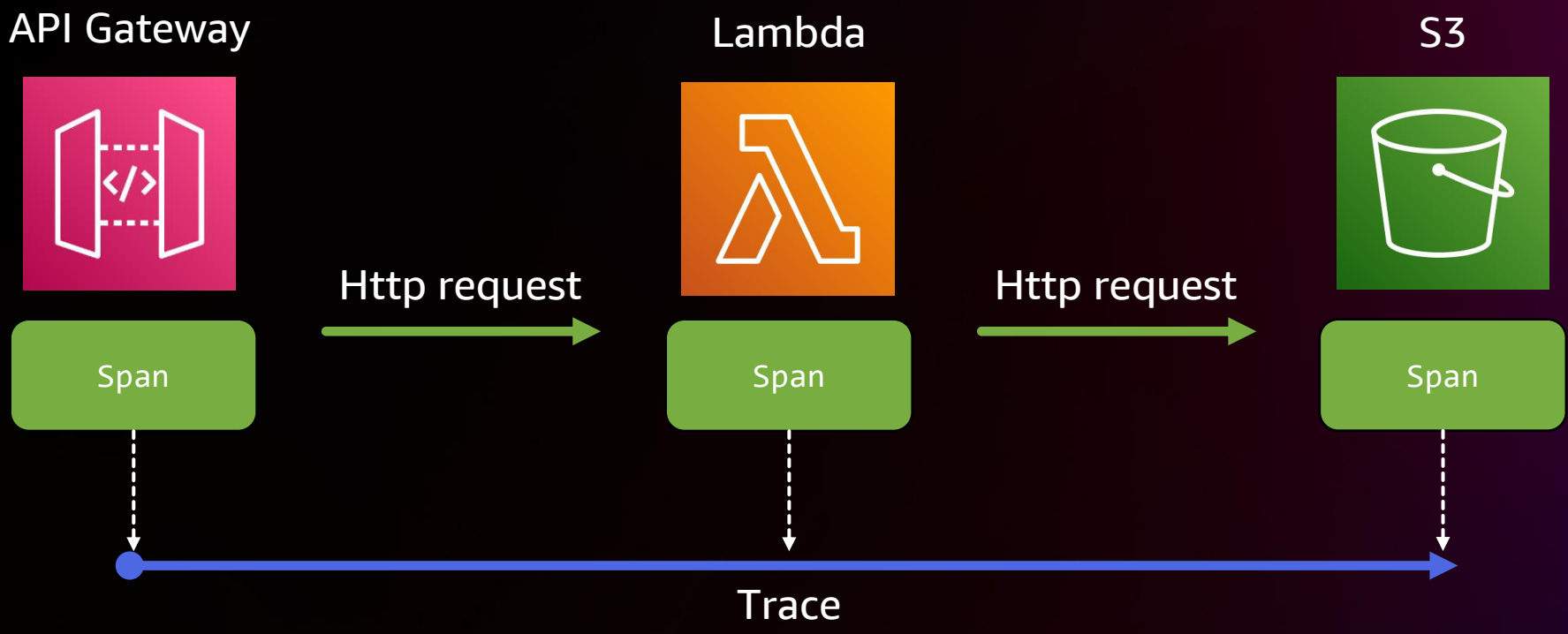
```
X-Amzn-Trace-Id:
    Root=1-5759e988-bd862e3fe1be46a994272793;
    Parent=53995c3f42cd8ad8;
    Sampled=1
```

Root=1-5759e988-bd862e3fe1be46a994272793;Parent=53995c3f42cd8ad8;Sampled=1 X-Amzn-Trace-Id: X-Ray hits adds a tracing header

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

Trace propagation

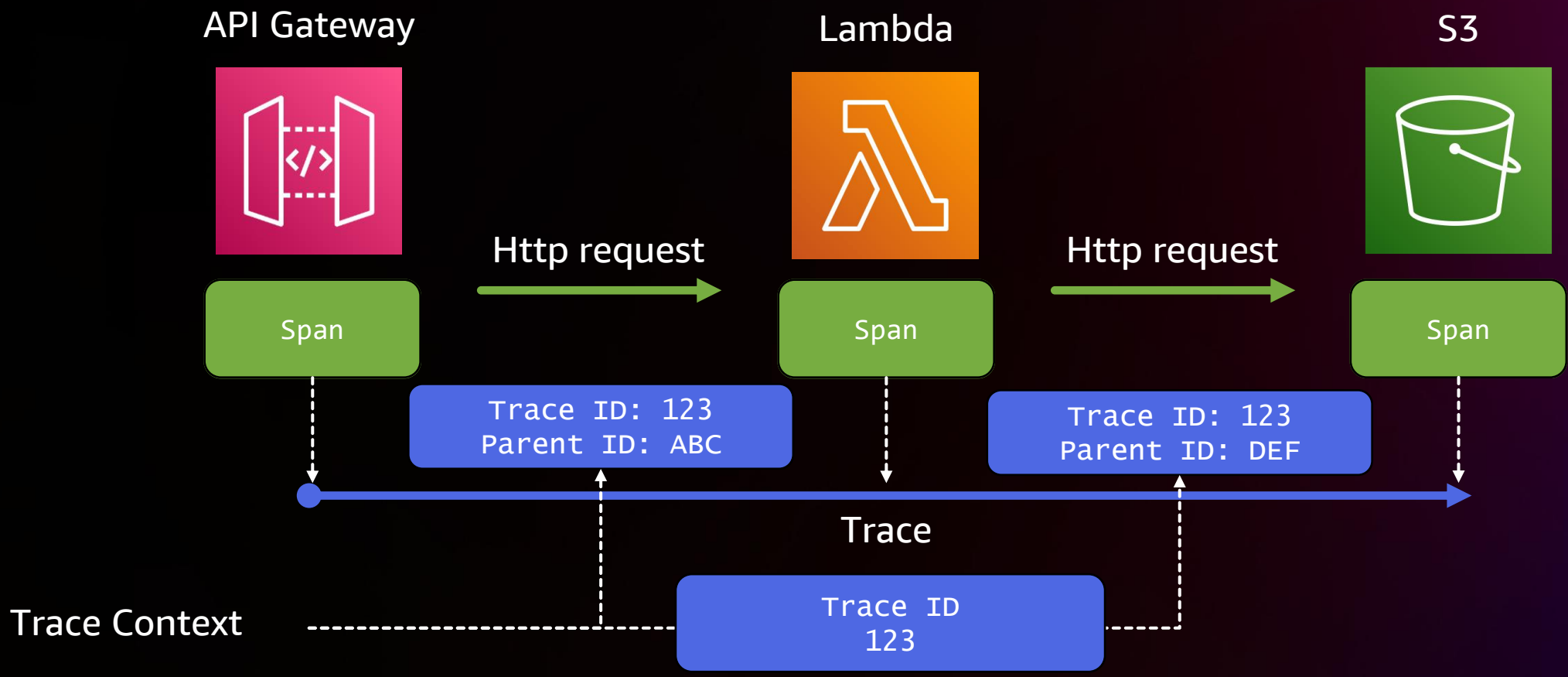
HOW DOES PROPAGATION WORK?



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

Trace propagation

HOW DOES PROPAGATION WORK?

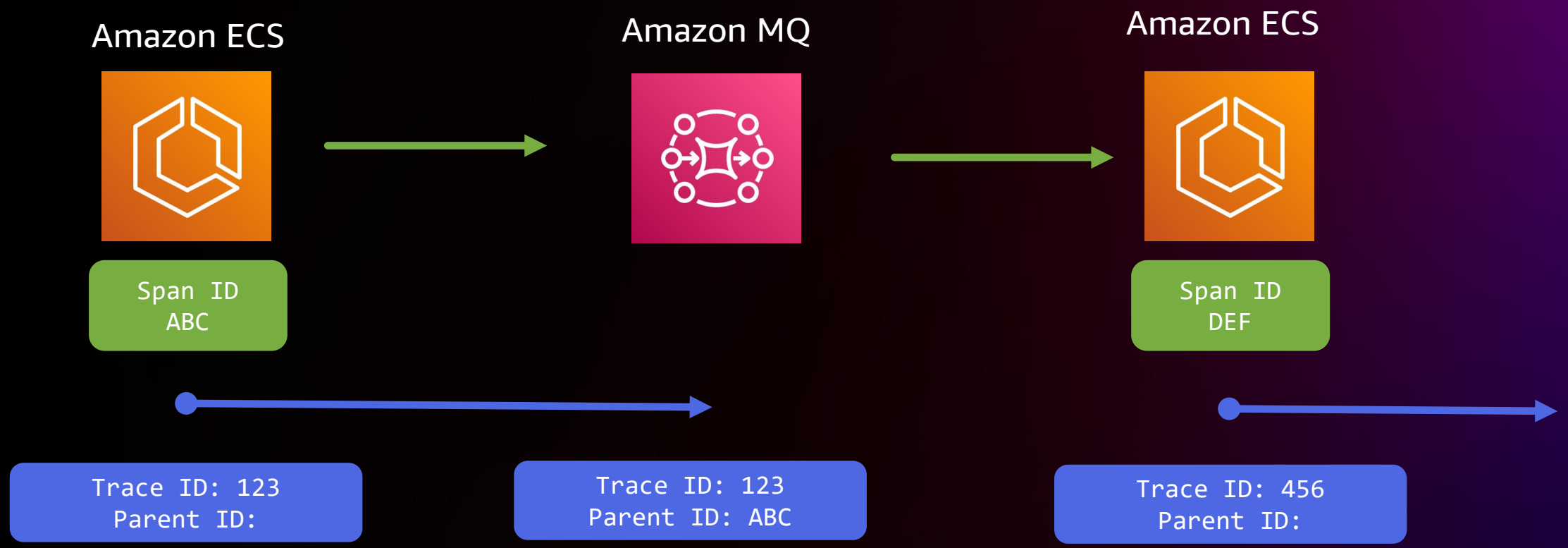


aws

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

Trace propagation

HOW DOES PROPAGATION WORK?



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

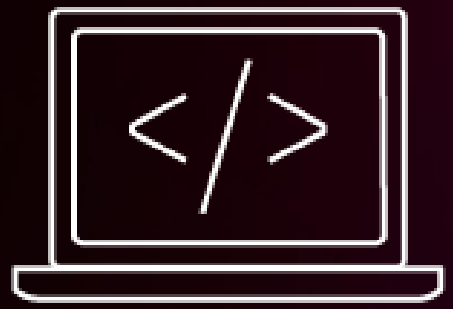
Getting started with tracing

Containers



Instrument your applications

Cloud Native



It depends

aws

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

Trace propagation

SERVICES THAT SUPPORT TRACING

API Gateway

App Mesh

App Runner

AWS AppSync

Amazon EC2

Elastic Load Balancing

CloudWatch RUM

EventBridge

Lambda

Amazon SNS

Step Functions

Amazon SQS

Amazon S3

CloudWatch Synthetics

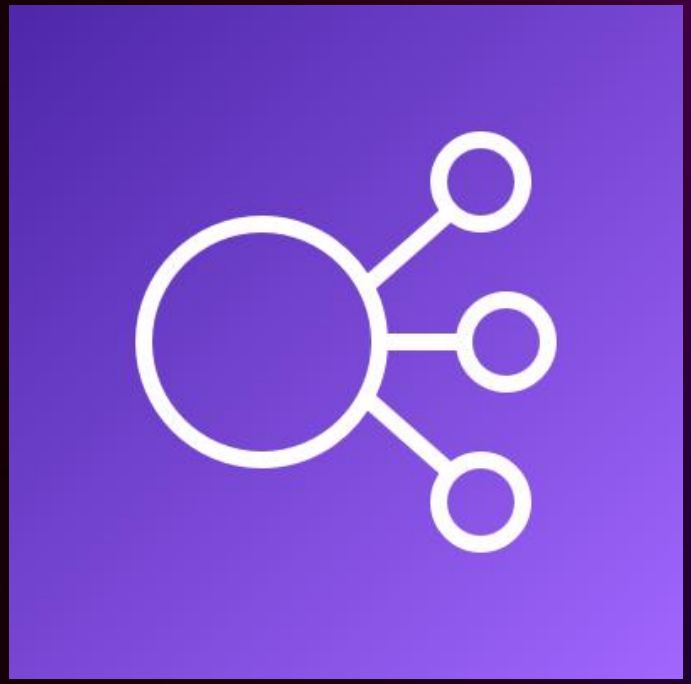


aws

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

Elastic Load Balancing

- Application load balancers add a trace ID to incoming HTTP requests
- Load balancers do not send data to X-Ray and do not appear as a node on your service map



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

Lambda

- Enabling X-Ray means Lambda sends traces to X-Ray automatically
- Instrumentation is required in your Lambda function to capture downstream calls
- X-Ray tracing is currently not supported for Lambda functions with Amazon MSK or Amazon MQ event source mappings



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

Amazon EventBridge

- If a service that is instrumented with the X-Ray SDK sends events to EventBridge, the trace context is propagated

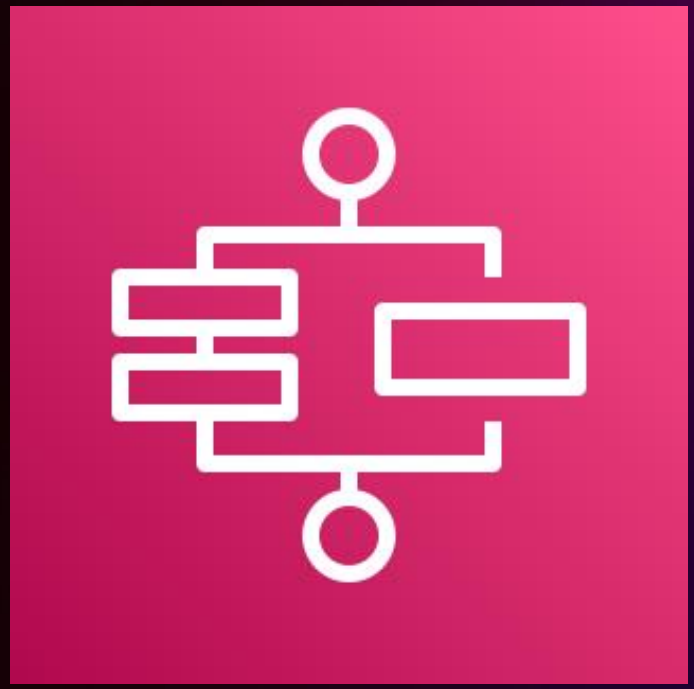
- EventBridge can pass a trace header for an event only if the event came from a PutEvents request that passed the trace context



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

AWS Step Functions

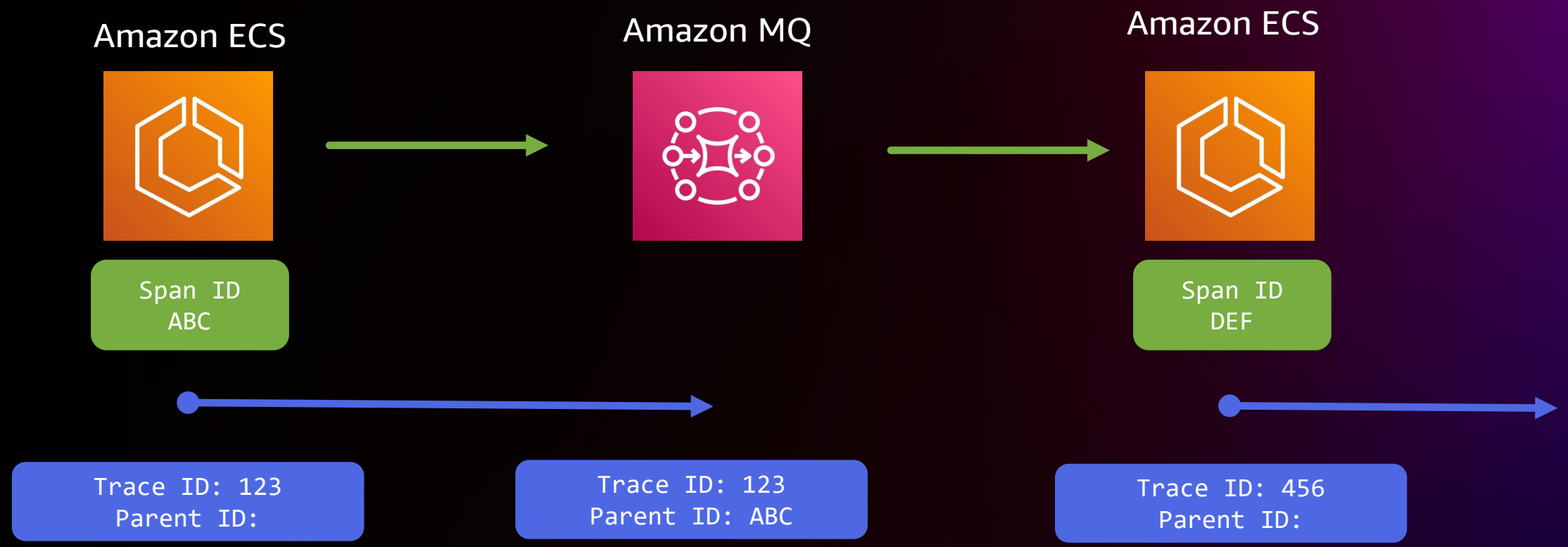
- With X-Ray enabled for your state machine, requests are traced as they are executed in Step Functions
- Amazon SNS, Amazon SQS, and AWS Lambda integrate with native X-Ray support
- Amazon ECS, AWS Batch, and AWS Fargate require instrumentation



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

Trace propagation

HOW DOES PROPAGATION WORK?



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

Trace propagation demo

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

Avoid dangling traces



SUMMARY

1. Instrument all your code

2. Understand which AWS services support tracing and how

3. If a service does not support tracing

   - Pass the trace context across the service boundary

   - Resume the context on the other side of the service boundary

aws

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

Summary



Navigating your instrumentation options



Optimizing the cost of high cardinality



Reducing alarm fatigue



Avoiding dangling traces

Start with OpenTelemetry for your instrumentation

Leverage both metrics and logs for high cardinality

Correlate and alarm on high-level metric anomalies

Ensure trace context propagation

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

Resources

One Observability Workshop



AWS Distro for OpenTelemetry



Observability Skill Builder



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

Thank you!

Roland Barcia
rolanbah@amazon.com

Greg Eppel
geppel@amazon.com



Please complete the session survey in the **mobile app**

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