Part 2

AWS Workshop Series

Day 5: Serverless

Taking Enterprise Beyond the Cloud by TrueIDC

Mr. Athiwat Itthiwatana

Cloud & Solution Consultant

Presented by

- Athiwat Itthiwatana (HAM)
- Cloud & Solution Consultant, TrueIDC
- AWS Specialist
- SAP Basis Specialist
- athiwat.itt@ascendcorp.com

trueIDC

Agenda

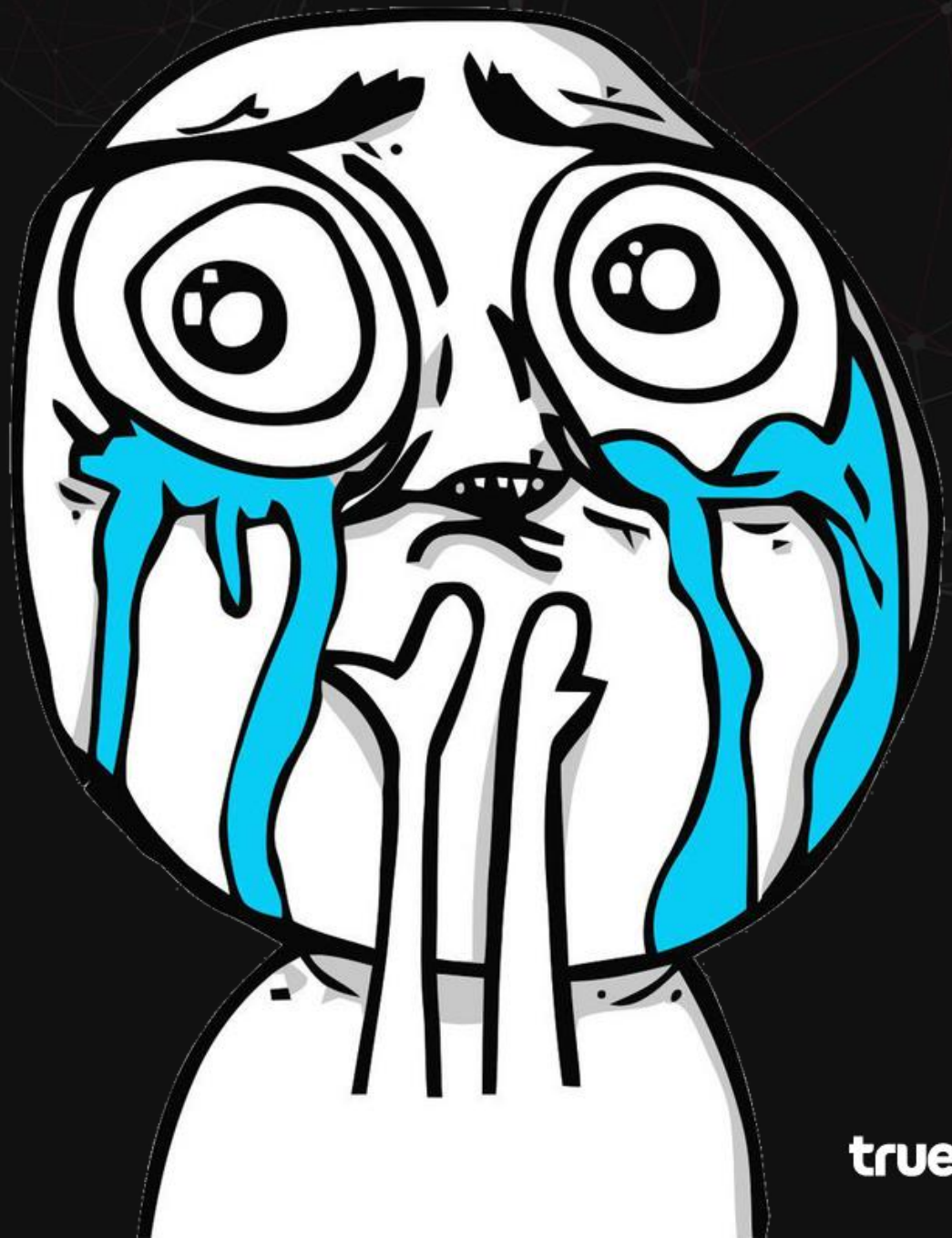
- Serverless Observability
- WildRydes Lab

Source: AWS Immersion day

Let’s get started with

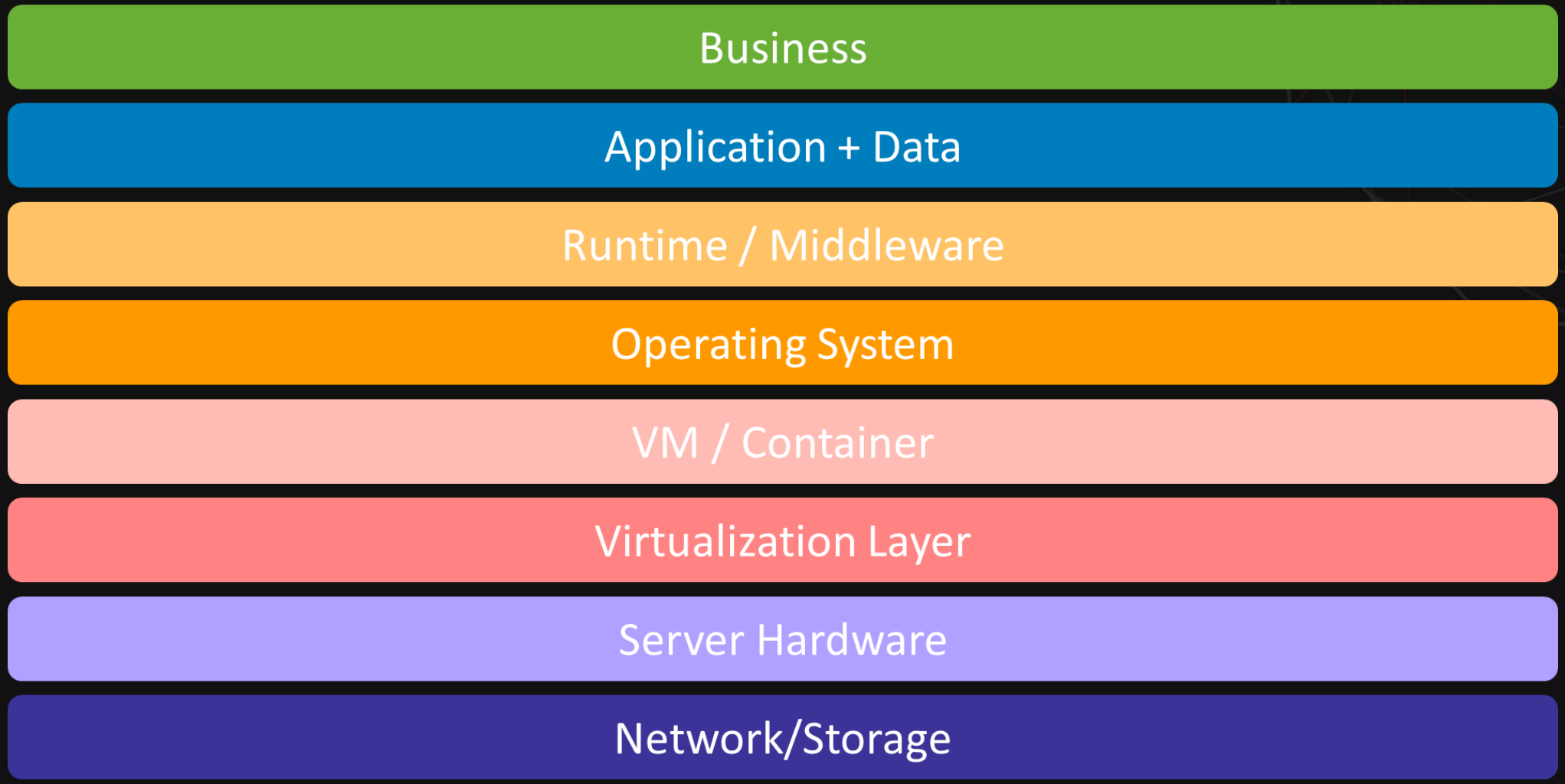
"Everything fails, all the time."

Dr. Werner Vogels,
Amazon CTO



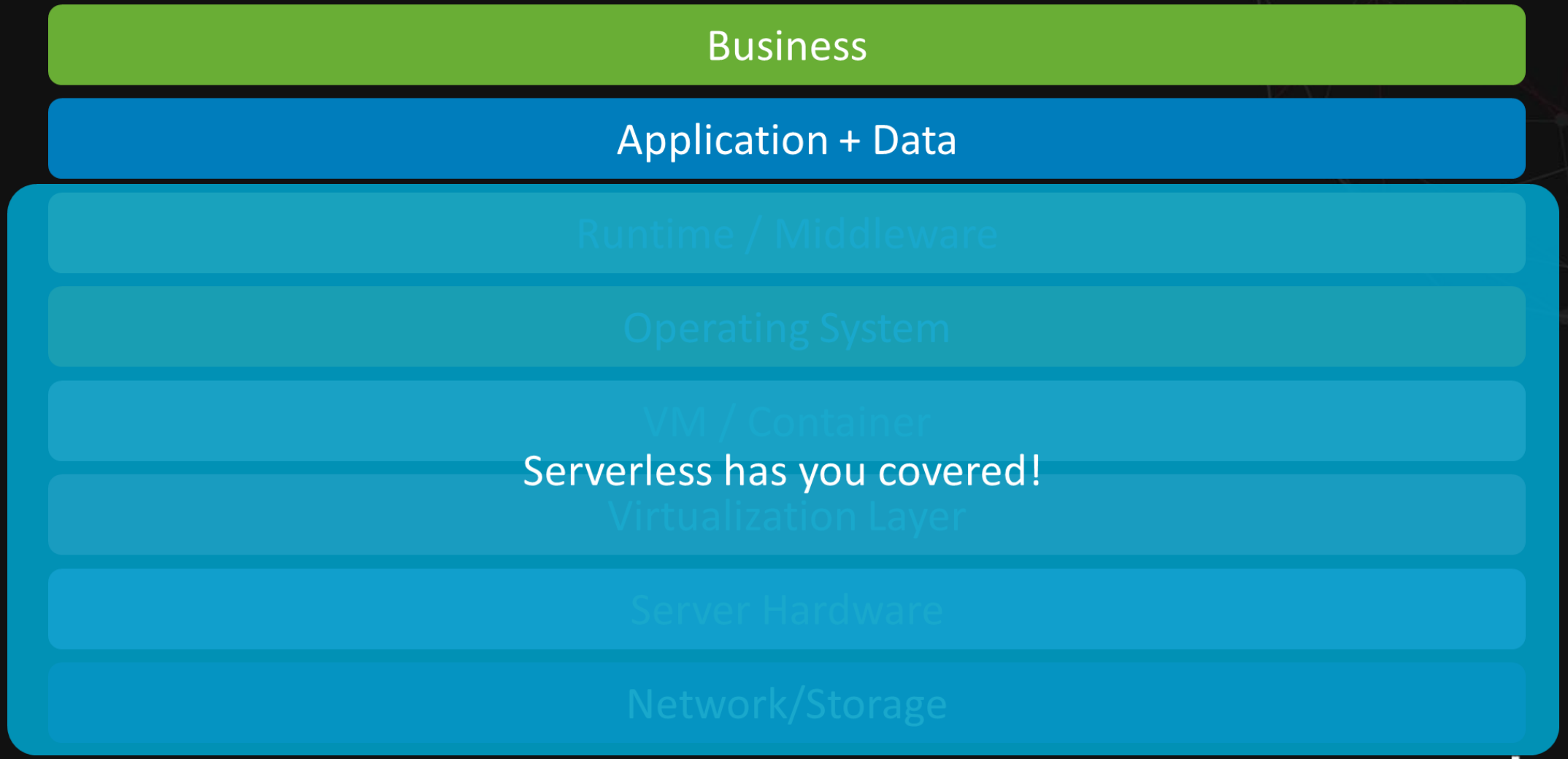
Source: AWS Immersion day

Traditional monitoring layers



Source: AWS Immersion day

Serverless monitoring layers



Source: AWS Immersion day

Monitoring more than failures

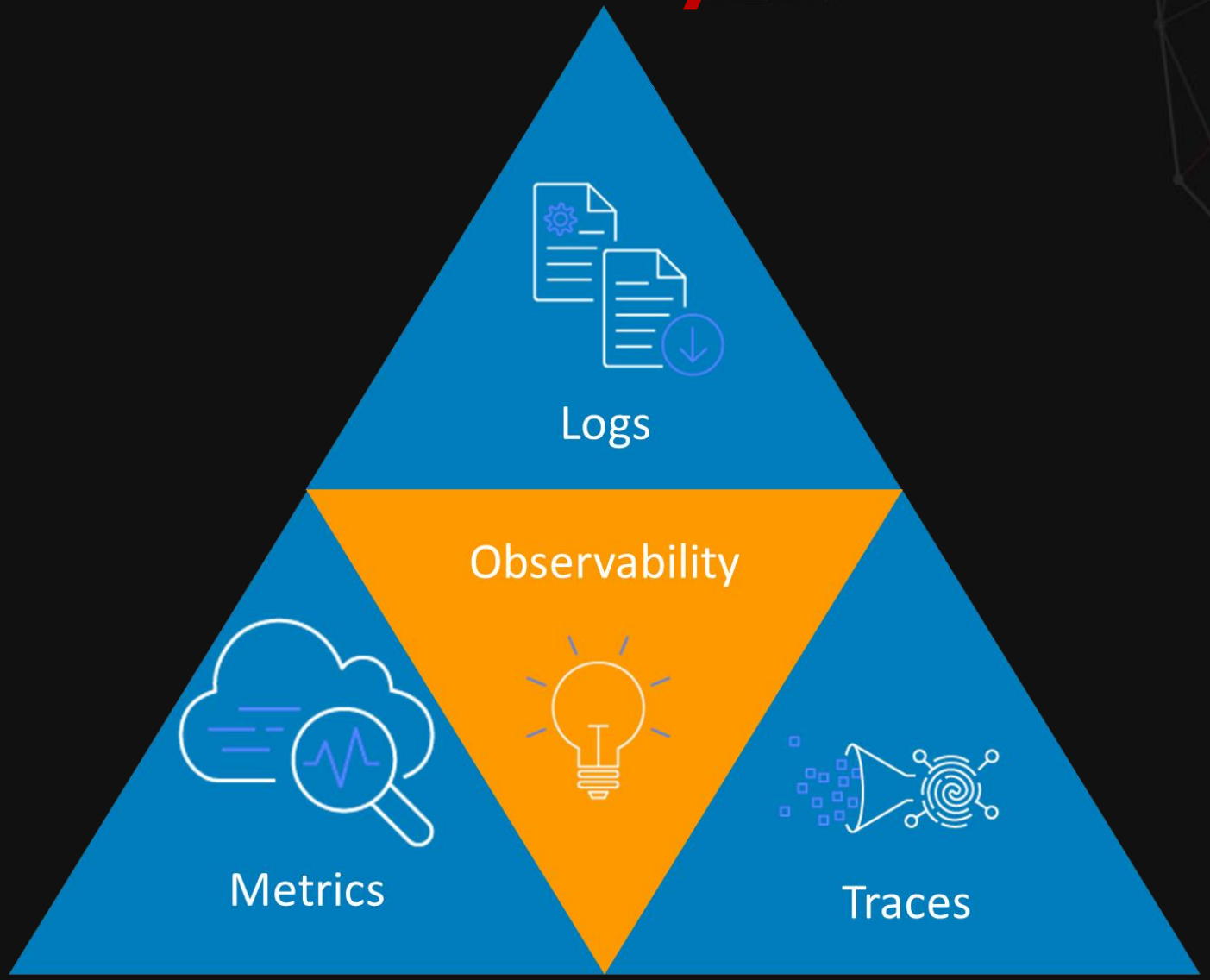
Is it behaving as expected?

What is the usage?

What is the business impact?

Source: AWS Immersion day

Serverless Observability



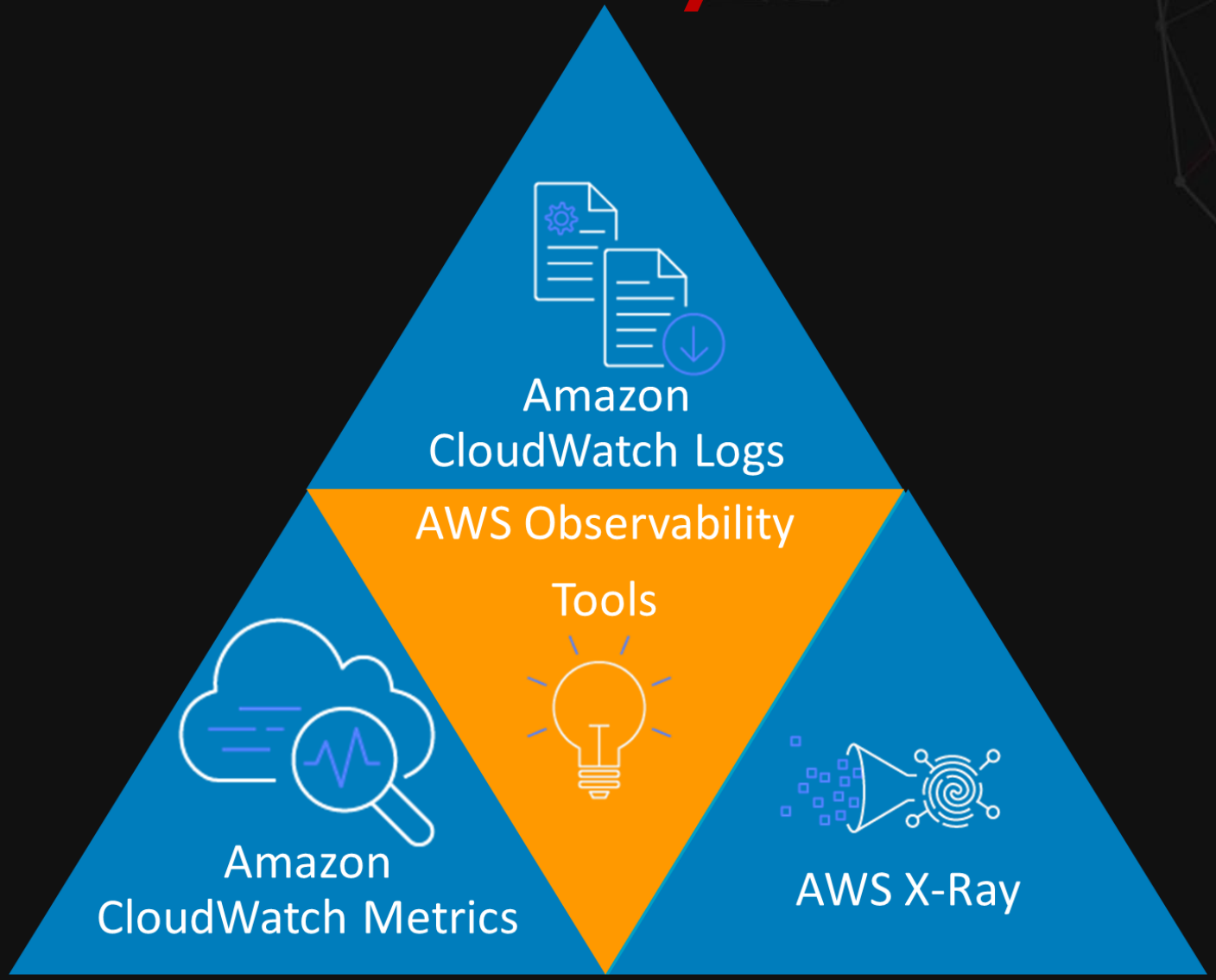
Source: AWS Immersion day

Three pillars of observability

Metrics	Logs	Traces
Numeric data measured at various time intervals (time series data); SLIs (request rate, error rate, duration, CPU%, etc.)	Timestamped records of discrete events that happened within an application or system, such as a failure, an error, or a state transformation	A trace represents a single user's journey across multiple applications and systems (usually microservices)

Source: AWS Immersion day

Serverless Observability

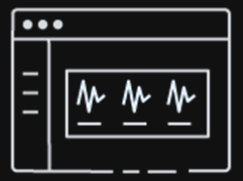


Source: AWS Immersion day

Breadth and depth of CloudWatch and X-Ray

Collect

- Embedded Metric Format
- Metric Filters
- StatsD & CollectD
- AWS PrivateLink



Monitor

- Cross-Account, Cross-Region Dashboards
- Automatic Dashboards
- Metric Math
- SQS and SNS add support for X-Ray

Act

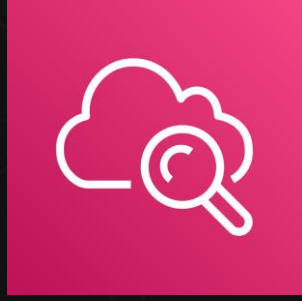
- Synthetics
- Anomaly Detection
- Metric Math Alarms
- Search Expressions

Analyze

- ServiceLens
- Contributor Insights
- Container Insights
- Logs Insights
- X-Ray Analytics

Source: AWS Immersion day

CloudWatch built-in metrics:



AWS Lambda

Invocation Metrics

Invocation Count, Invocation Errors, DeadLetterErrors, DestinationDeliveryFailures, Throttles, ProvisionedConcurrencyInvocations, ProvisionedConcurrencySpilloverInvocations

Performance Metrics

Duration, IteratorAge

Concurrency Metrics

ConcurrentExecutions, ProvisionedConcurrentExecutions, ProvisionedConcurrencyUtilization, UnreservedConcurrentExecutions

Amazon API Gateway

REST

API Calls Count, Latency, 4XXs, 5XXs, Integration Latency, Cache Hit Count, Cache Miss Count

HTTP

API Calls Count, Latency, 4XXs, 5XXs, Integration Latency, DataProcessed

WebSocket

Connect Count, Message Count, Integration Error, Client Error, Execution Error, Integration Latency

Source: AWS Immersion day

CloudWatch Embedded Metrics Format

Embed custom metrics alongside detailed log event data.

Automatically generate metrics from structured CloudWatch Logs.

Open-source client libraries available for Node.js and Python

Installation

```
npm install aws-embedded-metrics
```

Usage

To get a metric logger, you can either decorate

Using the metricScope decorator without funct

```
const { metricScope, Unit } = require("aws

const myFunc = metricScope(metrics =>
  async () => {
    metrics.putDimensions({ Service: "Aggr
    metrics.putMetric("ProcessingLatency",
    metrics.setProperty("RequestId", "422b
    // ...
  });

await myFunc();
```

Installation

```
pip3 install aws-embedded-metrics
```

Usage

To get a metric logger, you can decorate your function with a `metric_scope`:

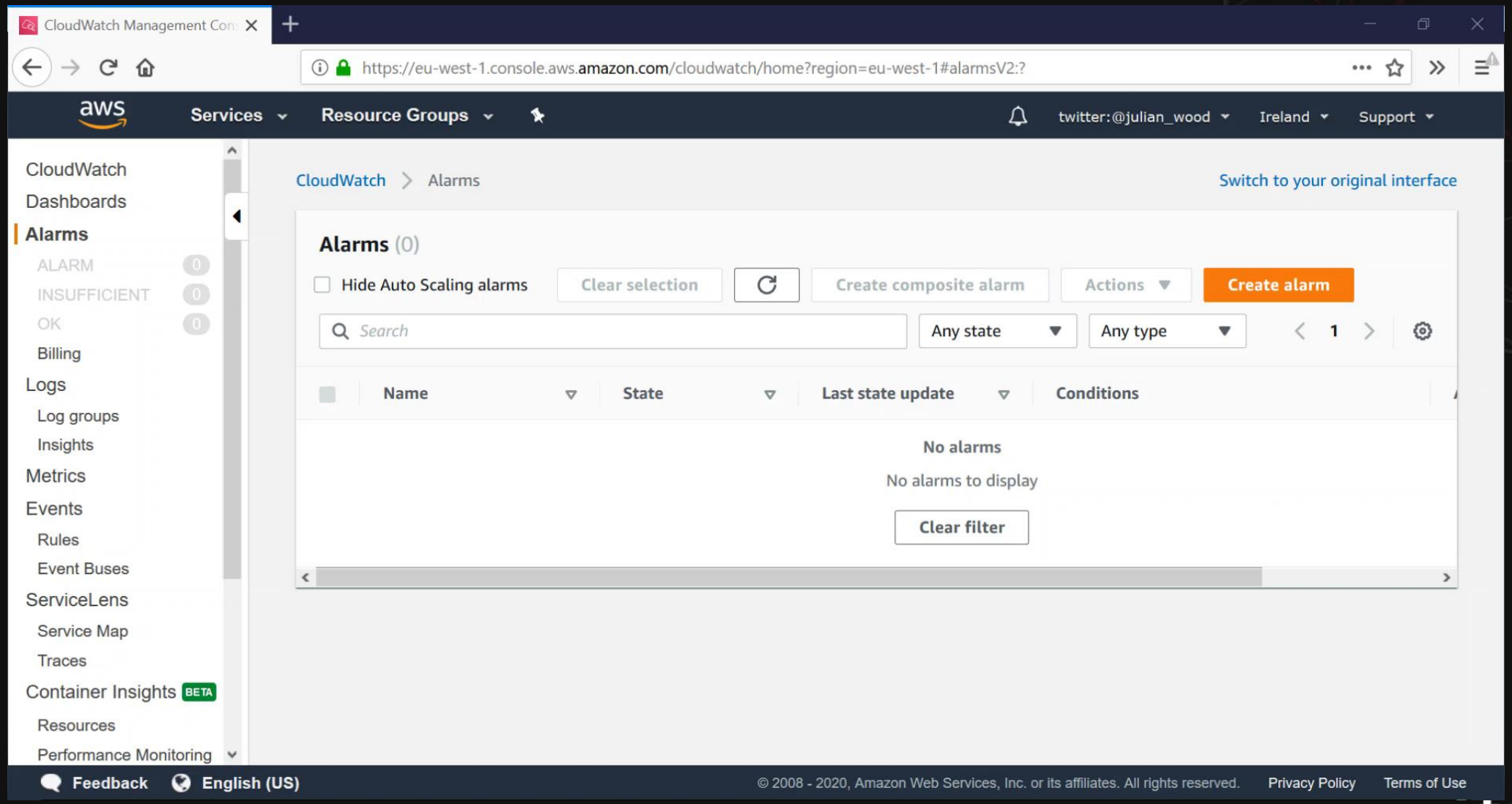
```
from aws_embedded_metrics import metric_scope

@metric_scope
def my_handler(metrics):
    metrics.put_dimensions({"Foo": "Bar"})
    metrics.put_metric("ProcessingLatency", 100, "Milliseconds")
    metrics.set_property("AccountId", "123456789012")
    metrics.set_property("RequestId", "422b1569-16f6-4a03")
    metrics.set_property("DeviceId", "61270781-c6ac-46f1")

    return {"message": "Hello!"}
```

Source: AWS Immersion day

Creating alerts



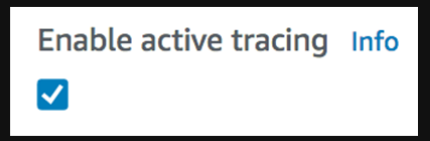
Source: AWS Immersion day

AWS X-Ray

End-to-end view of requests flowing through an application

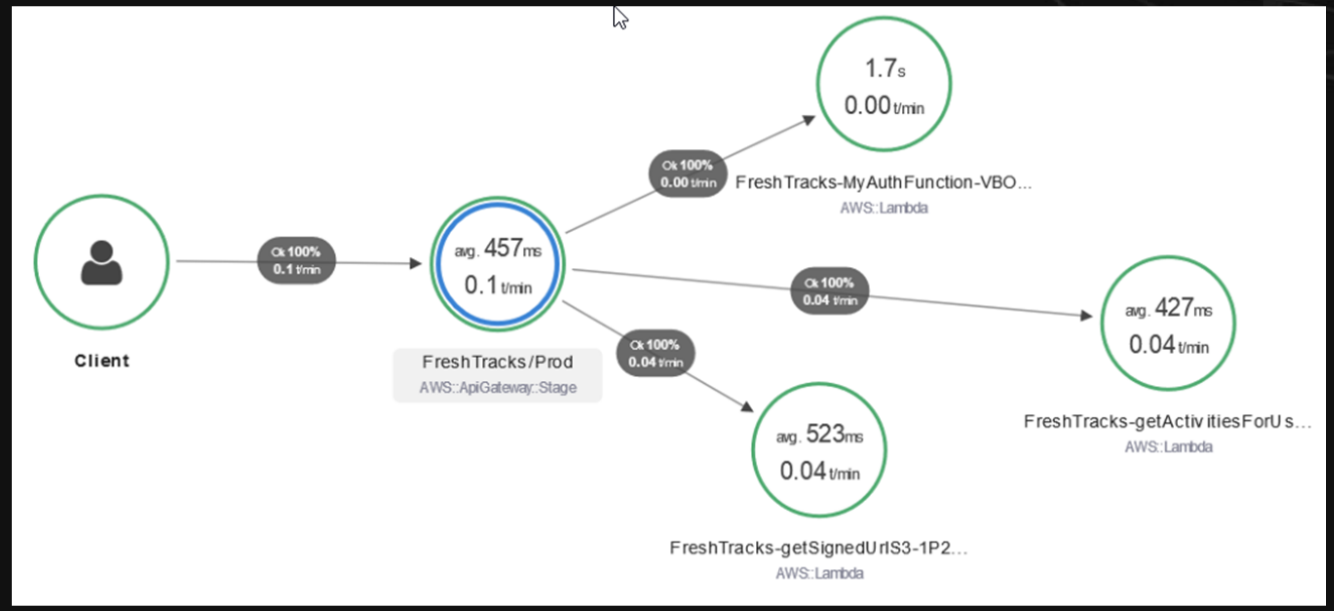
- Lambda: instruments incoming requests for all supported languages and can capture calls made in code

- API Gateway: inserts a tracing header into HTTP calls as well as reports data back to X-Ray itself




```
const AWSXRay = require('aws-xray-sdk-core');
const AWS = AWSXRay.captureAWS(require('aws-sdk'));

const documentClient = new AWS.DynamoDB.DocumentClient();
```



Source: AWS Immersion day

CloudWatch ServiceLens

Unified access to metrics, logs, traces and canaries.

Enabling performance monitoring from end-user interaction to infrastructure layer insights

Amazon CloudWatch

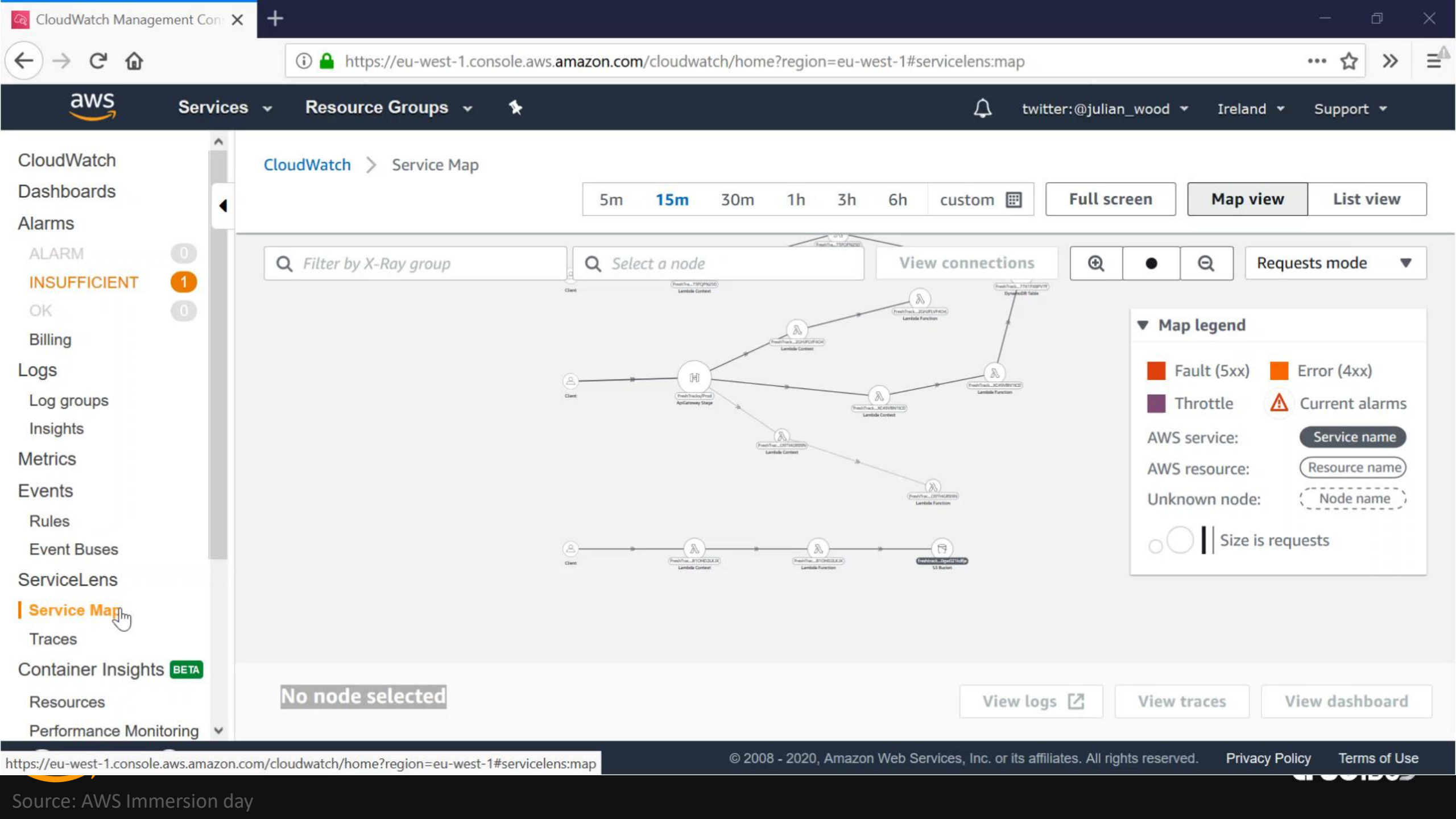
Metrics
Logs
Events
Alarms
Dashboards

AWS X-Ray

Traces
Analytics
Service Map
Latency detection
(server & client)

ServiceLens

Source: AWS Immersion day



Source: AWS Immersion day

Scenario: Wild Rydes

Source: AWS Immersion day

Help Wild Rydes Disrupt Transportation!

So how does this magic work?

DOWNLOAD THE APP

Head over to the app store and download the Wild Rydes app. You're just a few taps away from getting your ryde.

REQUEST A UNICORN

We can get you there. Simply request a ryde on the app and we'll connect you with a unicorn immediately.

PICK A PRICE

Pick the valuation you're willing to pay and your ryde is set up. The only surge is the acceleration you get when taking off.

RIDE OFF TO SUCCESS!

After matching with your unicorn and agreeing to its terms, you'll be all set. Your unicorn will arrive shortly to pick you up.

Your Task: Build the Wild Rydes Website

Welcome to Wild Rydes Inc.,
Employee #3!

aws

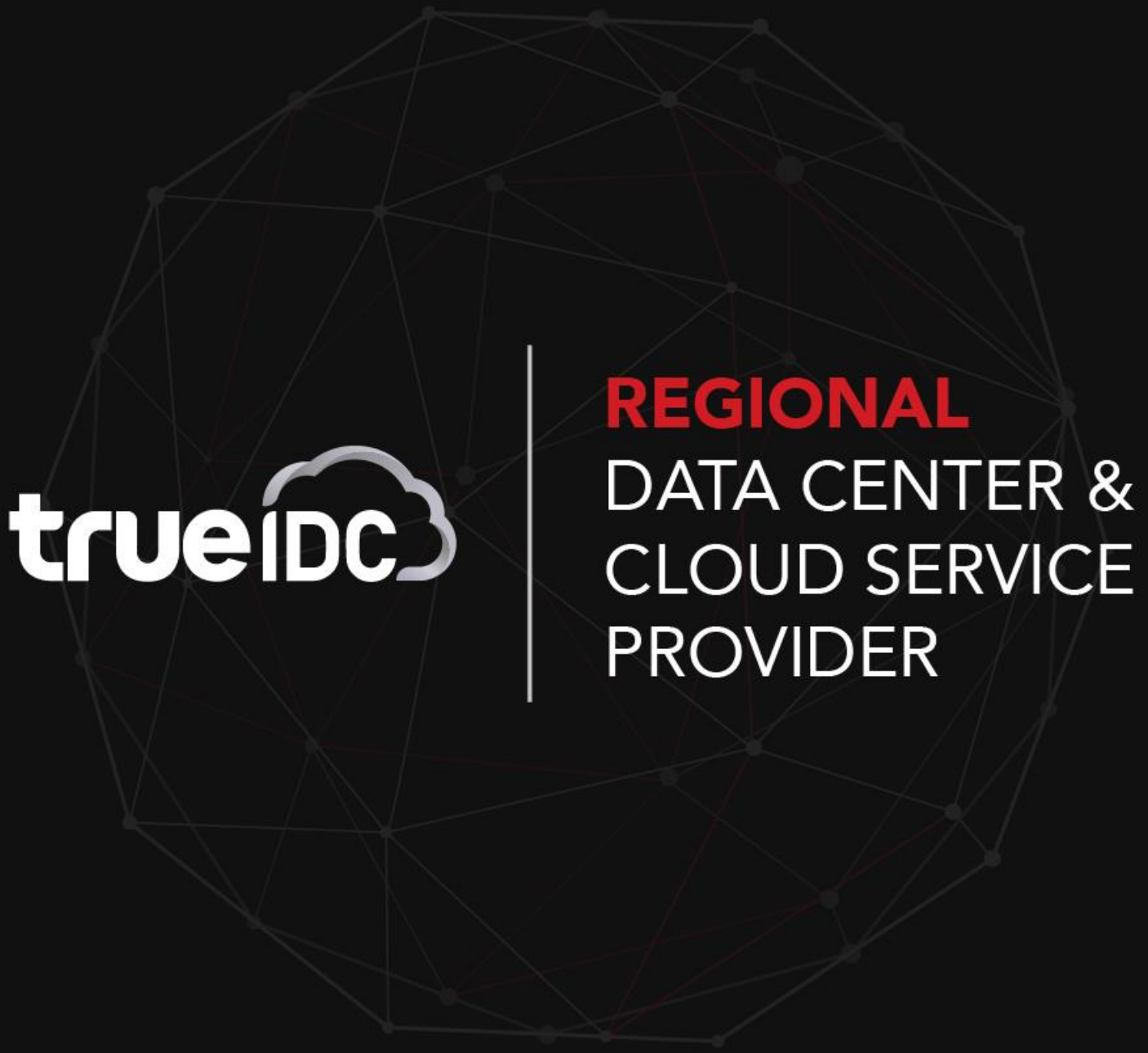
Source: AWS Immersion day

trueIDC

Lab: WILD RYDES

https://github.com/TIDC-PS-Inter/AWS-Workshop

trueIDC


trueIDC
REGIONAL
DATA CENTER &
CLOUD SERVICE
PROVIDER