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

SVS402-R

Serverless stream processing with AWS Lambda, Amazon Kinesis & Kafka

Veda Raman (she/her)

Senior AI/ML Specialist Solutions Architect
AWS

Adam Wagner (he/him)

Principal Serverless Specialist Solutions Architect
AWS

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

Agenda

Session expectations

Introduction to stream processing with AWS Lambda

Use case: Connected vehicle data

Use case: Real-time fraud detection

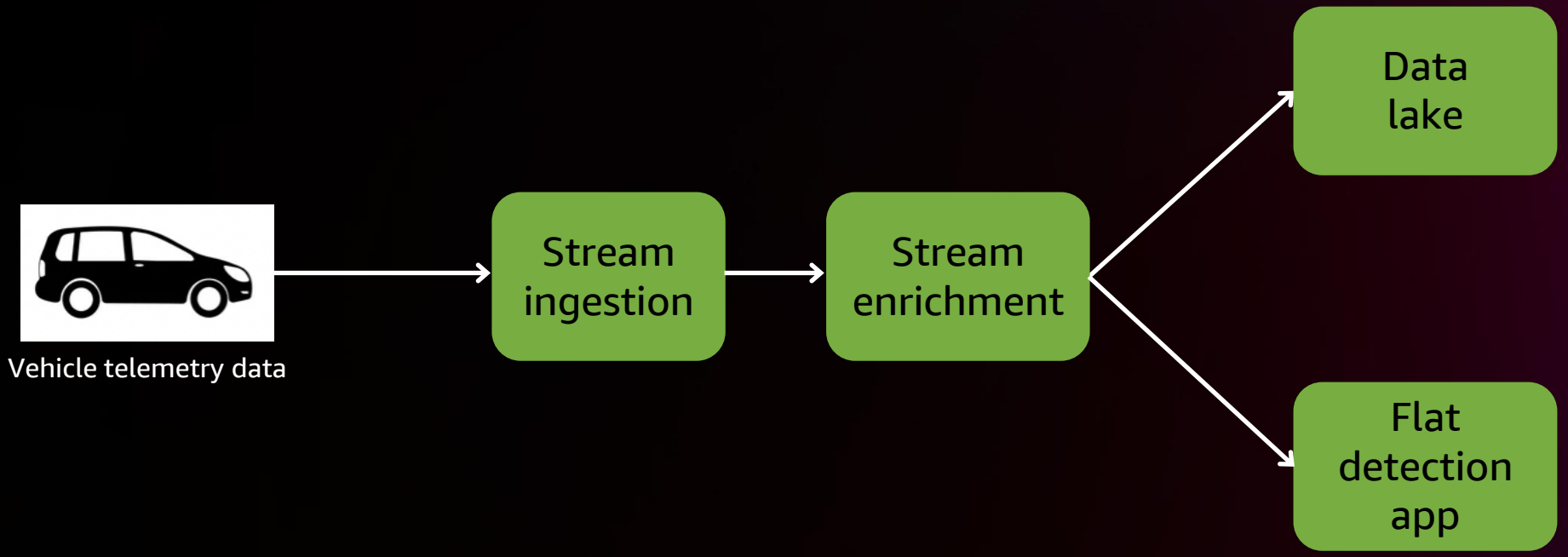
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Session expectations

- 400-level session – we are jumping into the deep end
- Chalk talk – please ask questions throughout

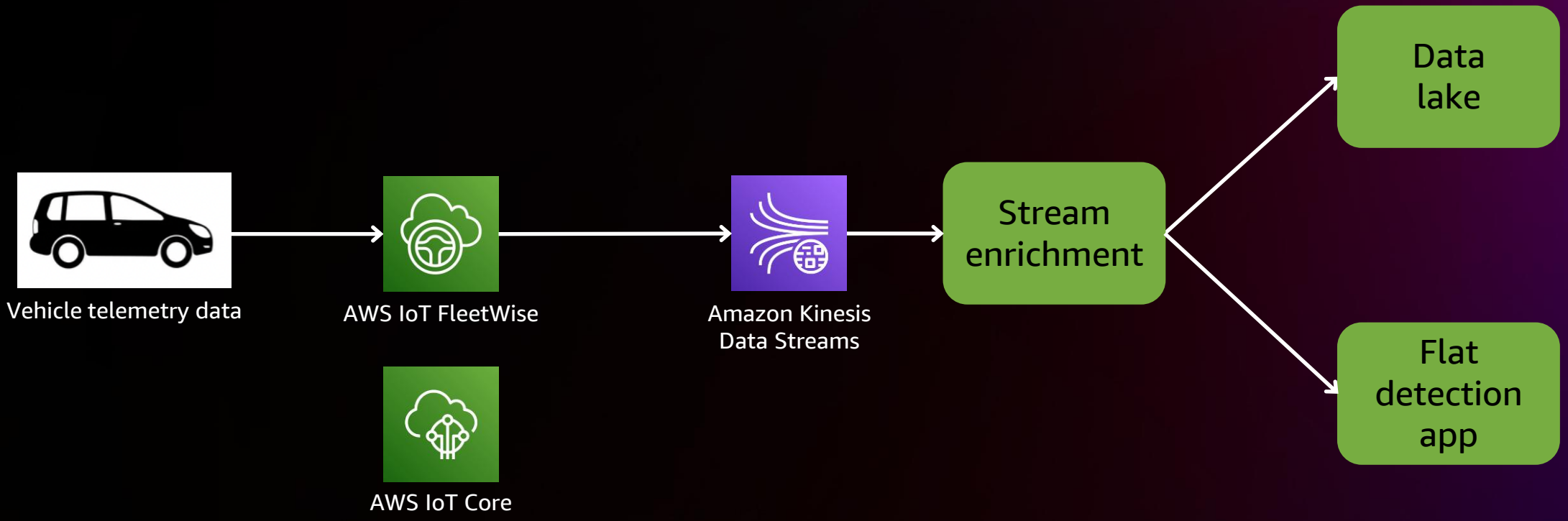
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Connected vehicle use case



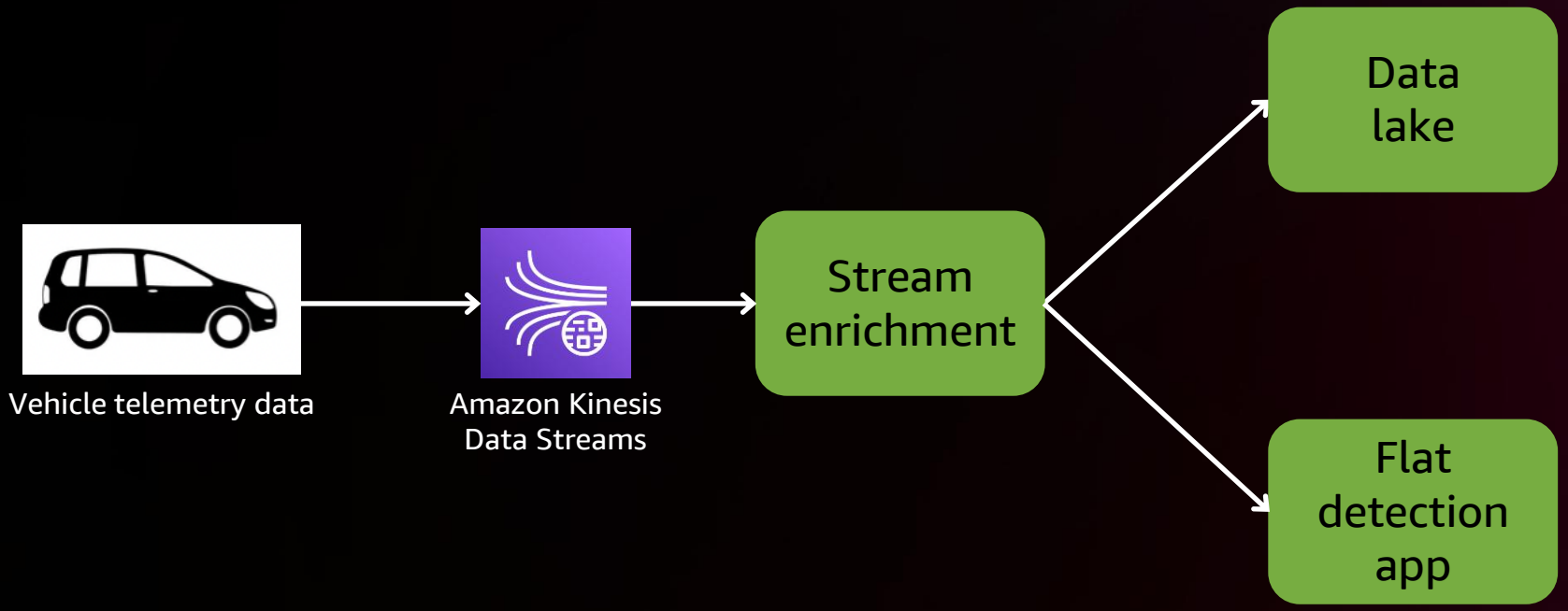
aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Connected vehicle use case



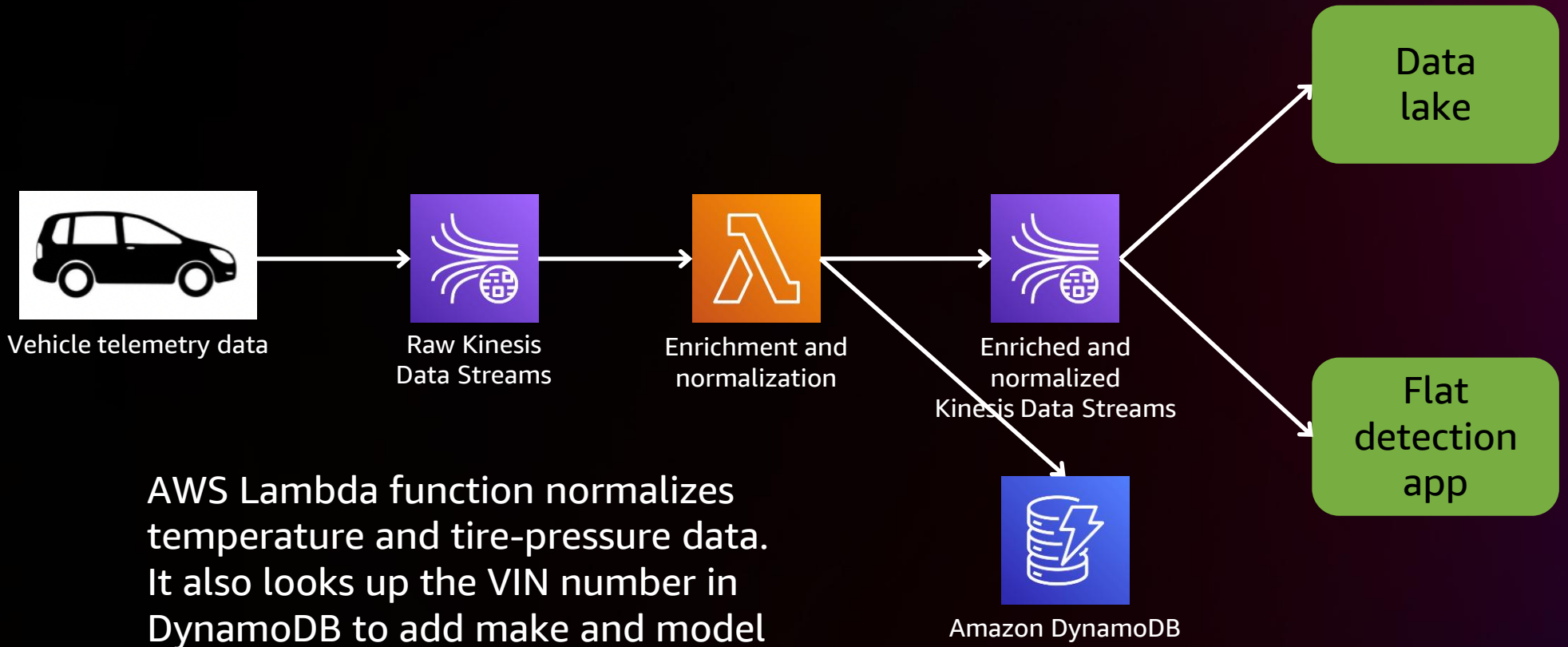
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Connected vehicle use case



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

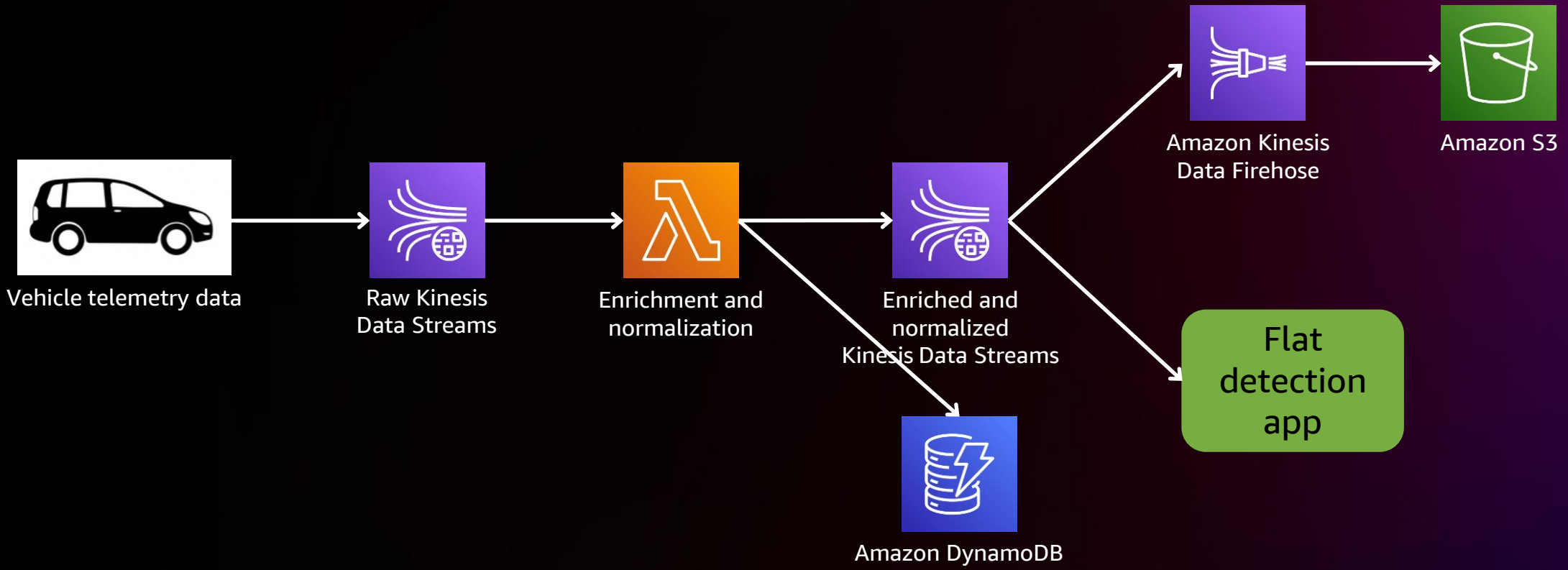
Connected vehicle use case



AWS Lambda function normalizes temperature and tire-pressure data. It also looks up the VIN number in DynamoDB to add make and model information.

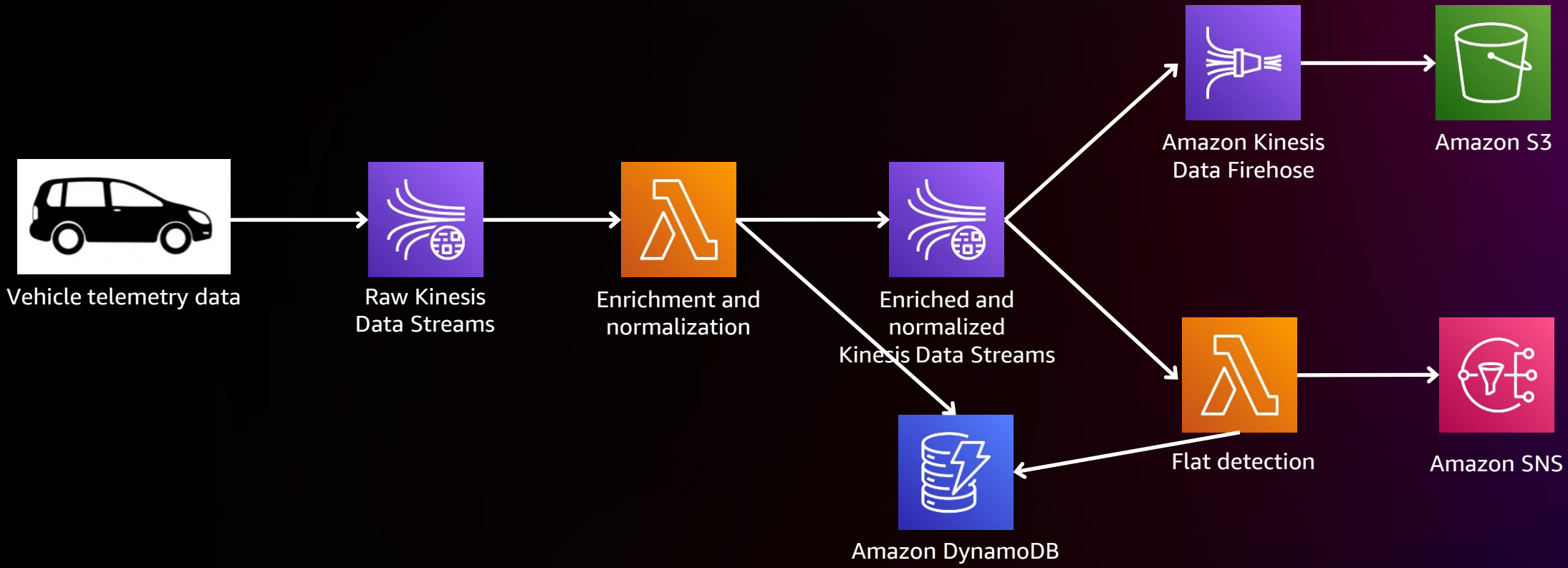
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Connected vehicle use case



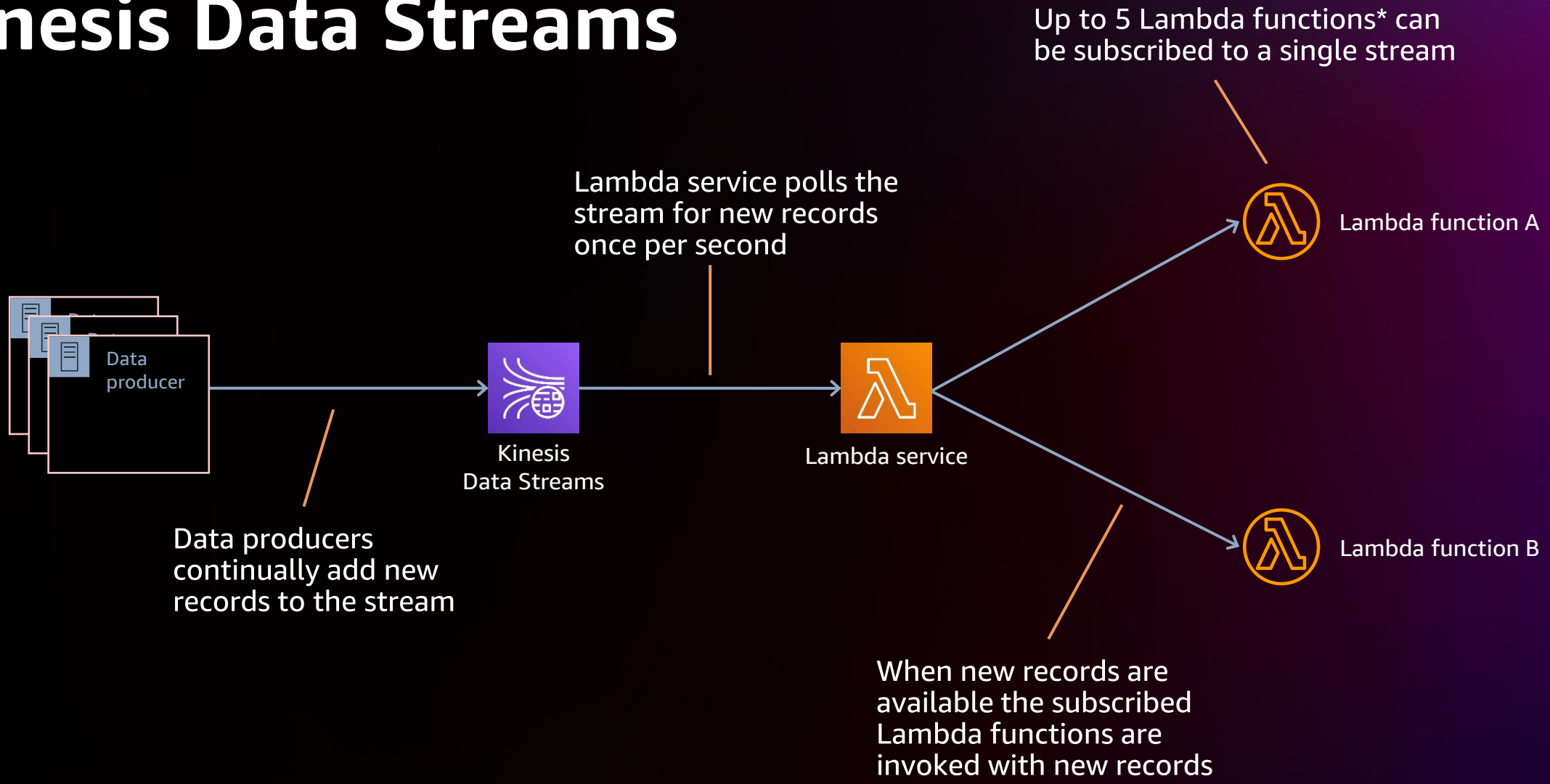
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Connected vehicle use case



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

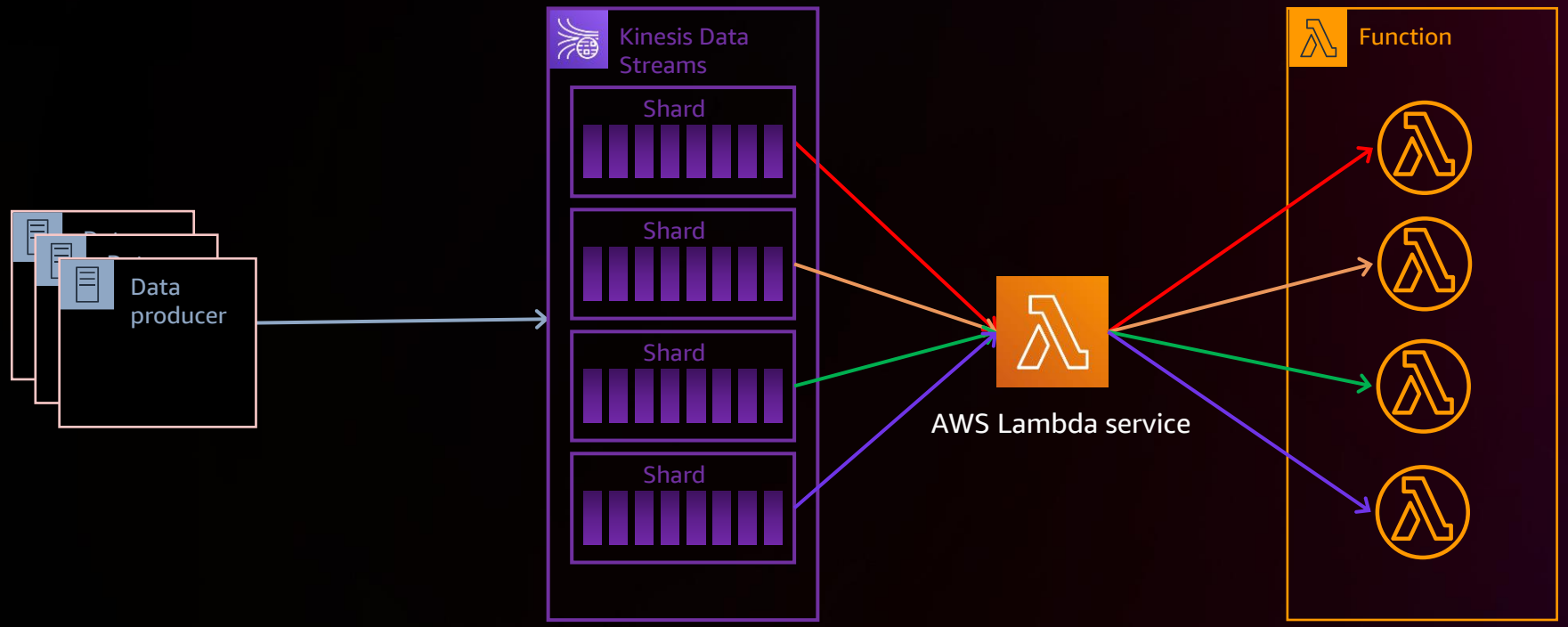
Kinesis Data Streams



* Amazon Kinesis Data Analytics and Kinesis Data Firehose also count toward this subscriber limit

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

Kinesis Data Streams shard detail



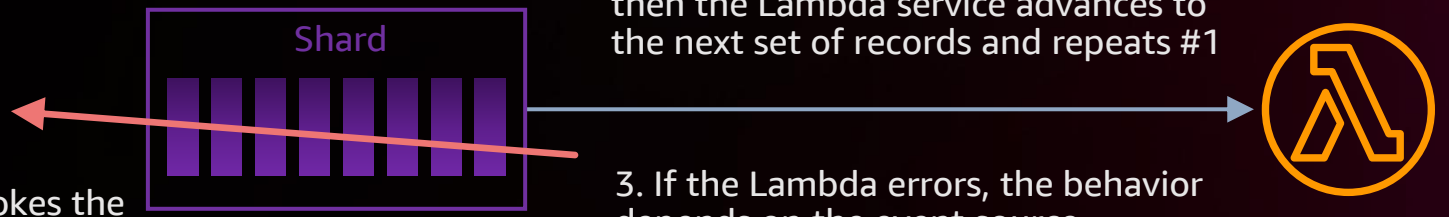
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Kinesis Data Stream shard level detail

1. Lambda service polls the shard once per second for a set of records. Then synchronously invokes the Lambda function with the batch of records

1a. If parallelization factor is >1, the Lambda service creates sub-batches, each of which then proceeds down the rest of this path

2. If the Lambda returns successfully, then the Lambda service advances to the next set of records and repeats #1



3. If the Lambda errors, the behavior depends on the event source configuration

Default settings

The Lambda service invokes the function with the same set of records and will continue to do so until it succeeds or the records age out of the stream

Report batch item failures

The Lambda function returns the record sequence number of the first failed item it encounters. The service advances its pointer to start the batch at that record and then proceeds with bisect batch and other error handling

Bisect batch on function error

The Lambda service splits the batch and retries the older half first. It continues to do this until it finds the problematic record. Once it can no longer split the batch, the single problematic record is processed based on the maximum retry and record age settings. Note that earlier retries as part of the bisecting do NOT count toward maximum retries

Maximum retries and maximum age

The Lambda service retries a batch until either the records are older than the maximum record age, or it has reached the maximum retry number. Once these are reached the records are sent to the configured failure destination. If no failure destination is configured, they are discarded

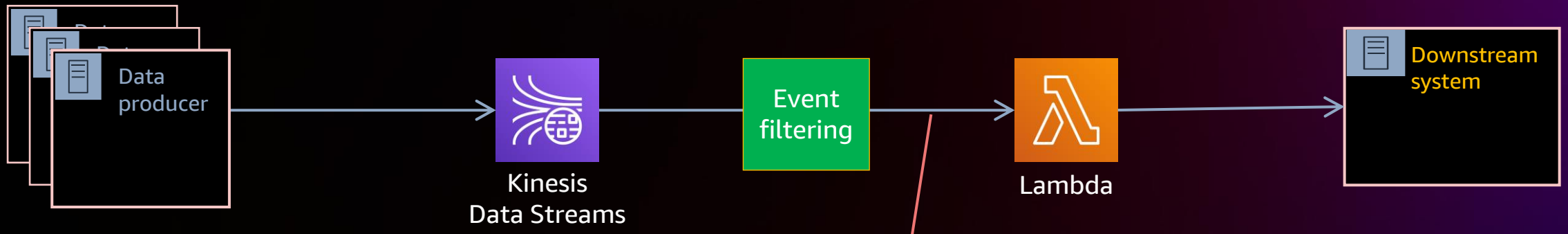
aws

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

Efficiency

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

AWS Lambda: Event filtering for Kinesis, Kafka, and Amazon DynamoDB Streams



Batch size and batch window are applied after the filtering

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

AWS Lambda: Event filtering

- Filter incoming messages **before function invocation**
  - Same pattern matching rules as Amazon EventBridge
  - Define up to 5 patterns per event source mapping, up to 2048 characters each
  - Batching applied **after filtering**
- Reduce traffic to function, **reduce cost**
- Support for: Kinesis Data Streams, DynamoDB Streams, Amazon SQS, Amazon MSK, self-managed Kafka

```
FleetTirePressureMapping:
    Type: AWS::Lambda::EventSourceMapping
    Properties:
      FunctionName: fleet-tire-pressure-evaluator
      BatchSize: 100
      StartingPosition: LATEST
      EventSourceArn:

        arn:aws:kinesis:...:stream/fleet-telemetry
      FilterCriteria:
        Filters:
          - Pattern: |
            {
              "data": {
                "tire_pressure": [{"numeric": ["<", 32]}]
              }
            }
```

Blog post

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

Performance

Measuring performance

Individual Lambdas vs. end-to-end testing

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

Performance

Scalability

- Kinesis on demand
- DDB on demand
- Kinesis scaling with Lambda

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

Performance step by step

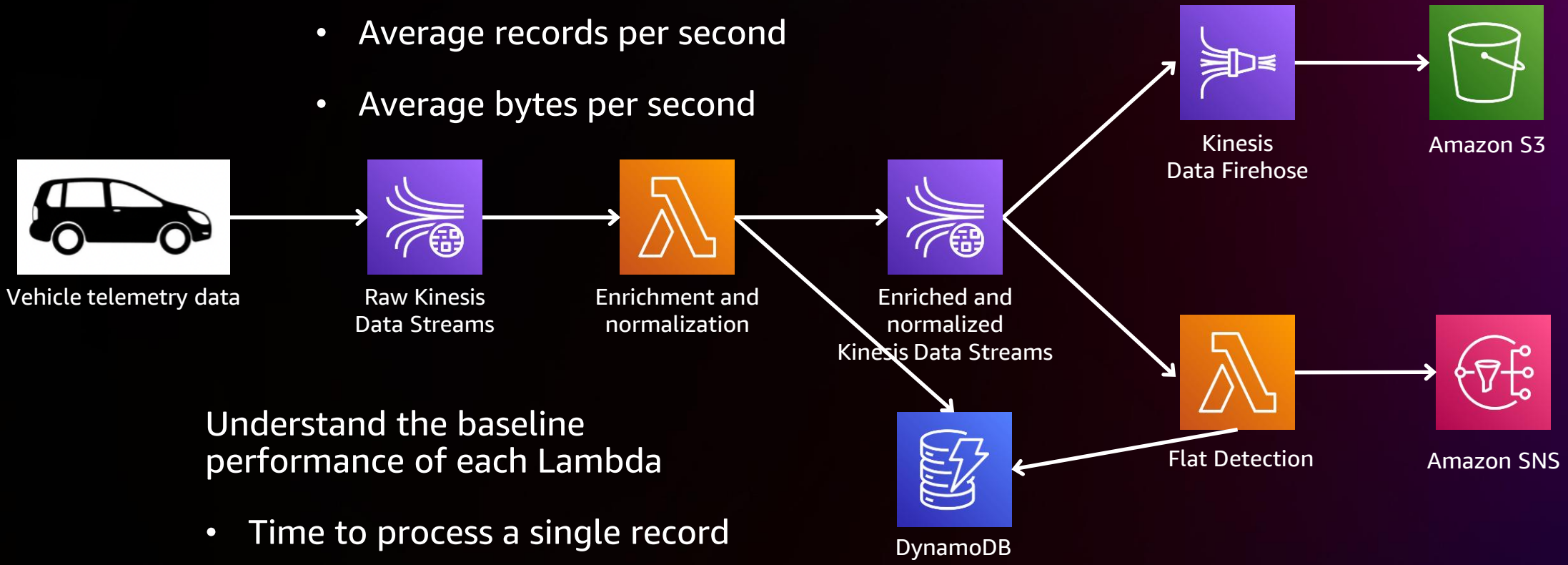
1. Ensure you have a working system
2. Measure performance
    1. Time to process 1 message
    2. Time to process average batch size
3. Tune the producer
4. Tune the Lambda function
    1. Tune the code
    2. Powertune the Lambda function with a representative batch
5. Tune the event source mapping
    1. Batch size and batch window
    2. Enhanced fan-out
    3. Parallelization factor
    4. Error handling

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

Performance measurement

Understand the baseline load on each Kinesis Data Stream

- Average records per second
- Average bytes per second



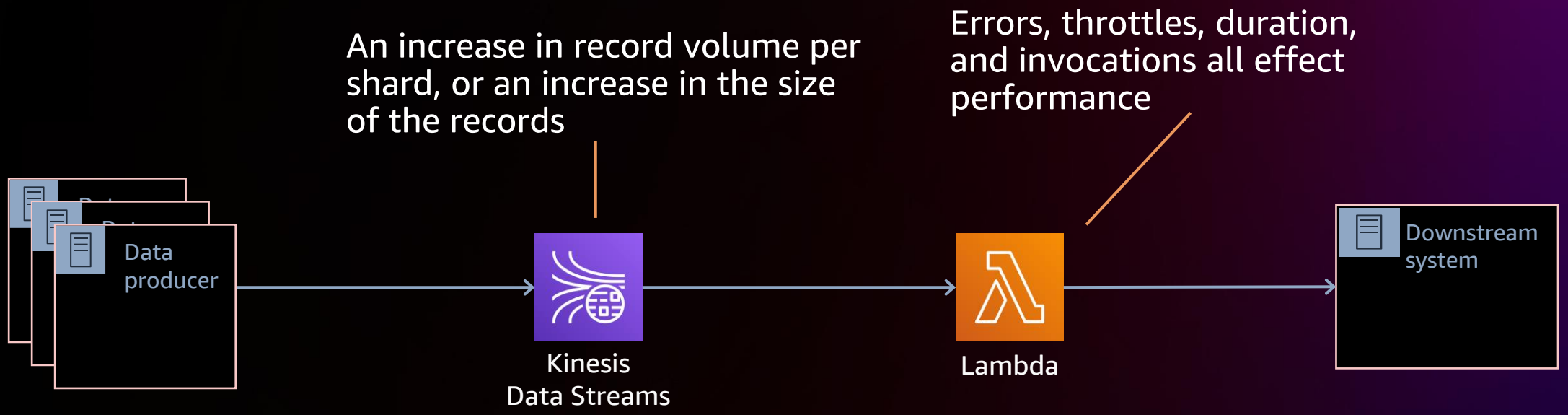
Understand the baseline performance of each Lambda

- Time to process a single record
- Time to process an average sized batch

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

Measuring performance

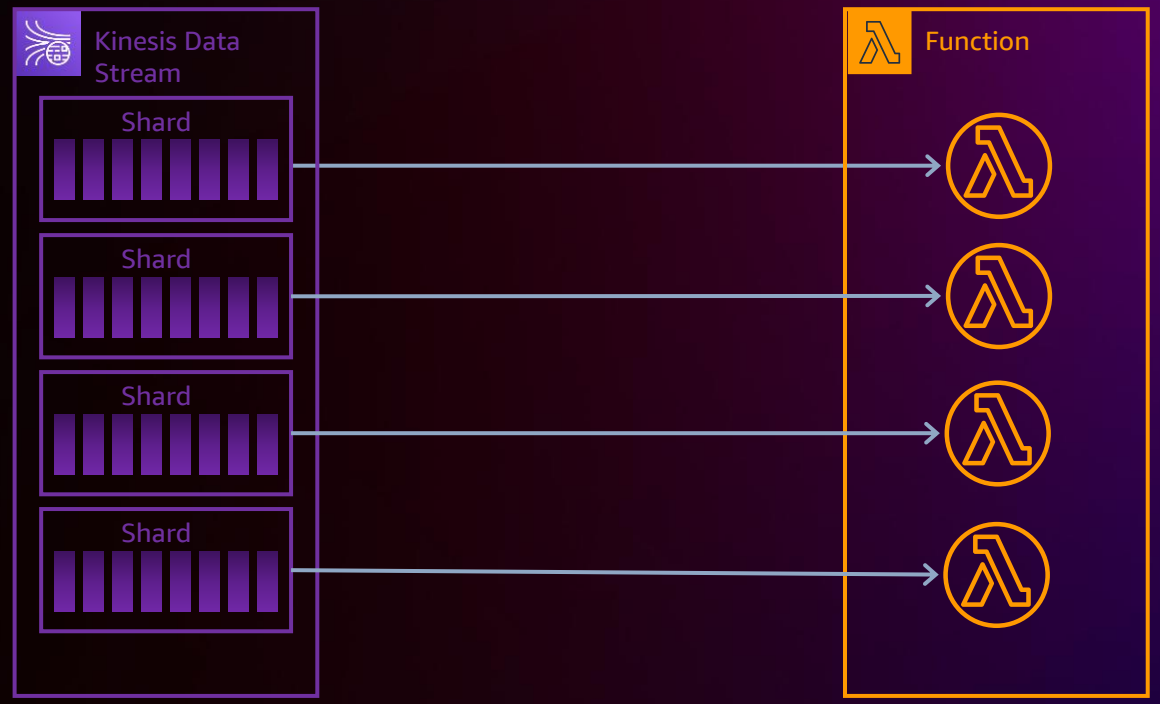
- Measuring performance
- Individual Lambdas vs. end-to-end testing



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

Parallelization factor

- Adds Lambda parallelization per shard
- Setting of 1 is the same as the current behavior, maximum setting is 10
- Batching via partition keys to maintain in order processing per partition key
- Works with both Kinesis Data Streams and DynamoDB Streams




```
--parallelization-factor 1
```

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

Parallelization factor

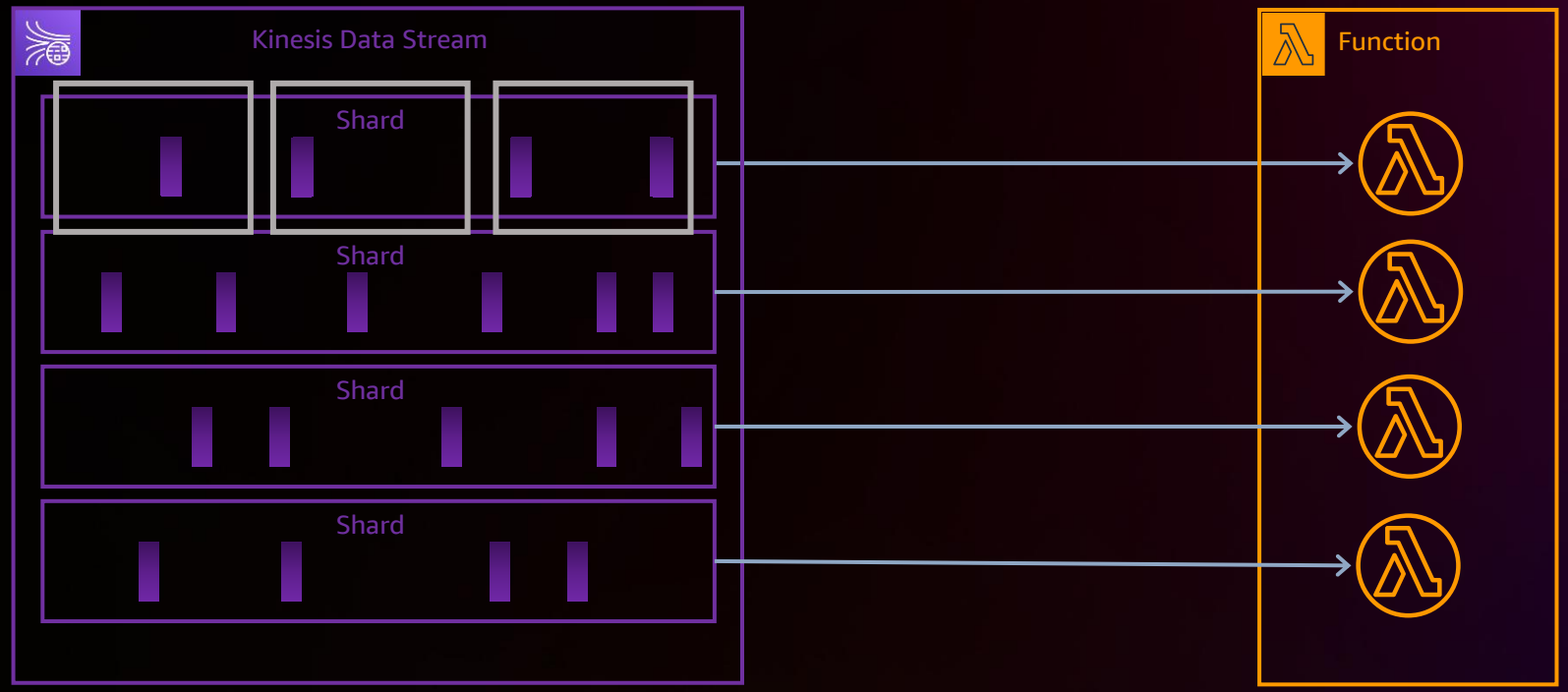
- Adds Lambda parallelization per shard
- Setting of 1 is the same as the current behavior, maximum setting is 10
- Batching via partition keys to maintain in order processing per partition key
- Works with both Kinesis Data Streams and DynamoDB Streams



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

Low throughput streams

- Lambda triggered with very small batches
- Leads to higher cost per message
- For archiving workloads the resulting payload is too small



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

Batch window

- Additional knob to tune the event source mapping
  - Set a time to wait before triggering. Max 5 minutes, set in seconds.
  - Batch size is still respected and will trigger on full batches before the batch window is up
  - Works for Kinesis Data Streams, Kafka, and DynamoDB event source mappings



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

Reliability

- Error handling

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

What is Kafka?



Is it like a service bus?



Is It like REST but async?



Is it like a database?

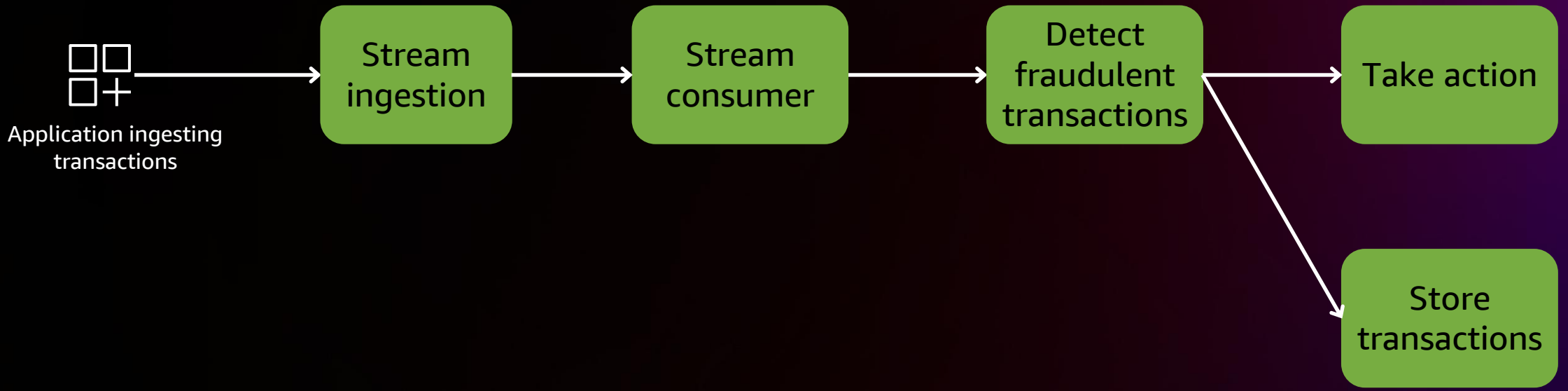


Streaming platform

aws

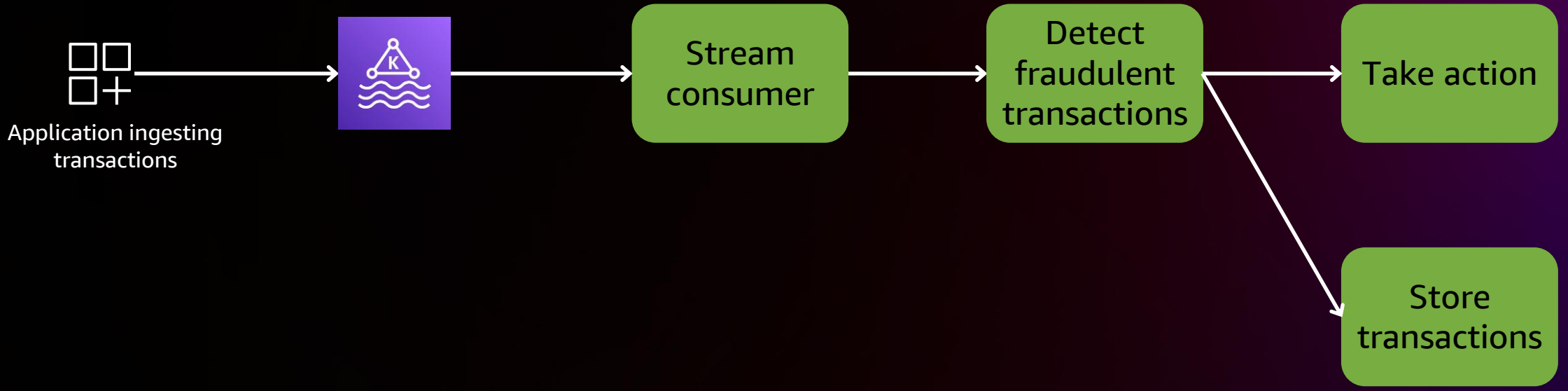
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Real-time transaction fraud detection



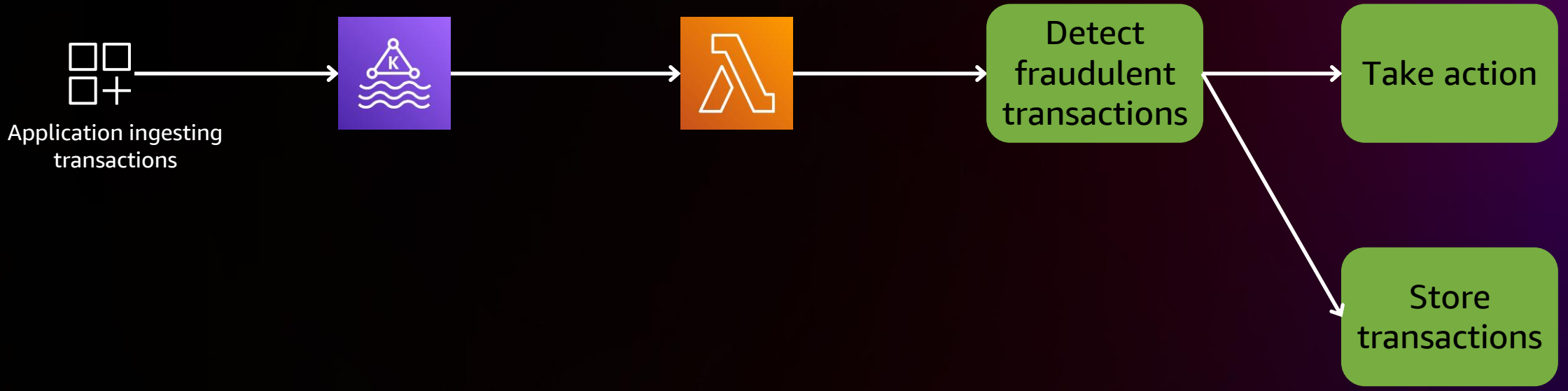
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Real-time transaction fraud detection



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

Real-time transaction fraud detection



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

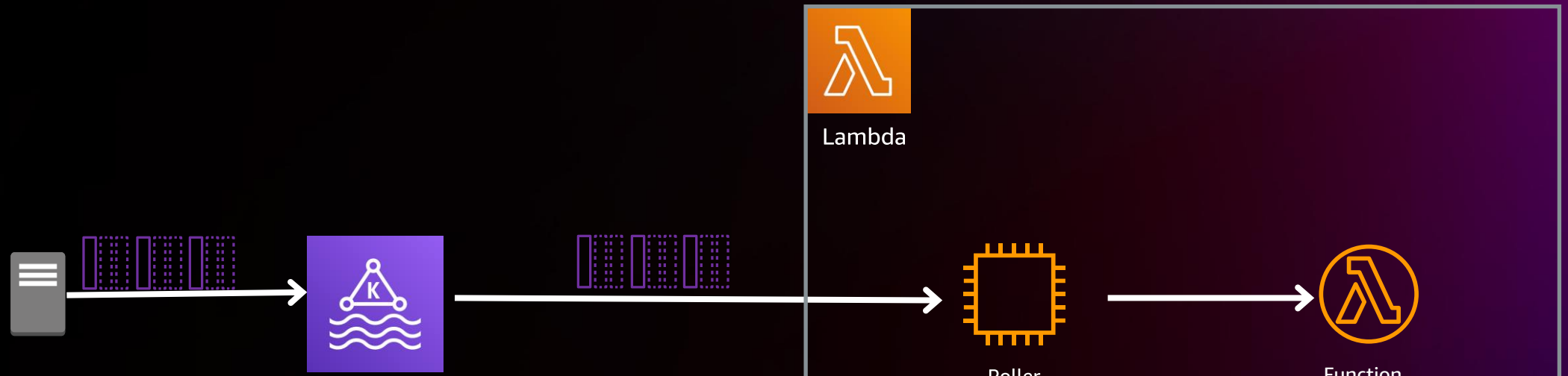
Real-time fraud detection using Kafka (Amazon MSK)





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

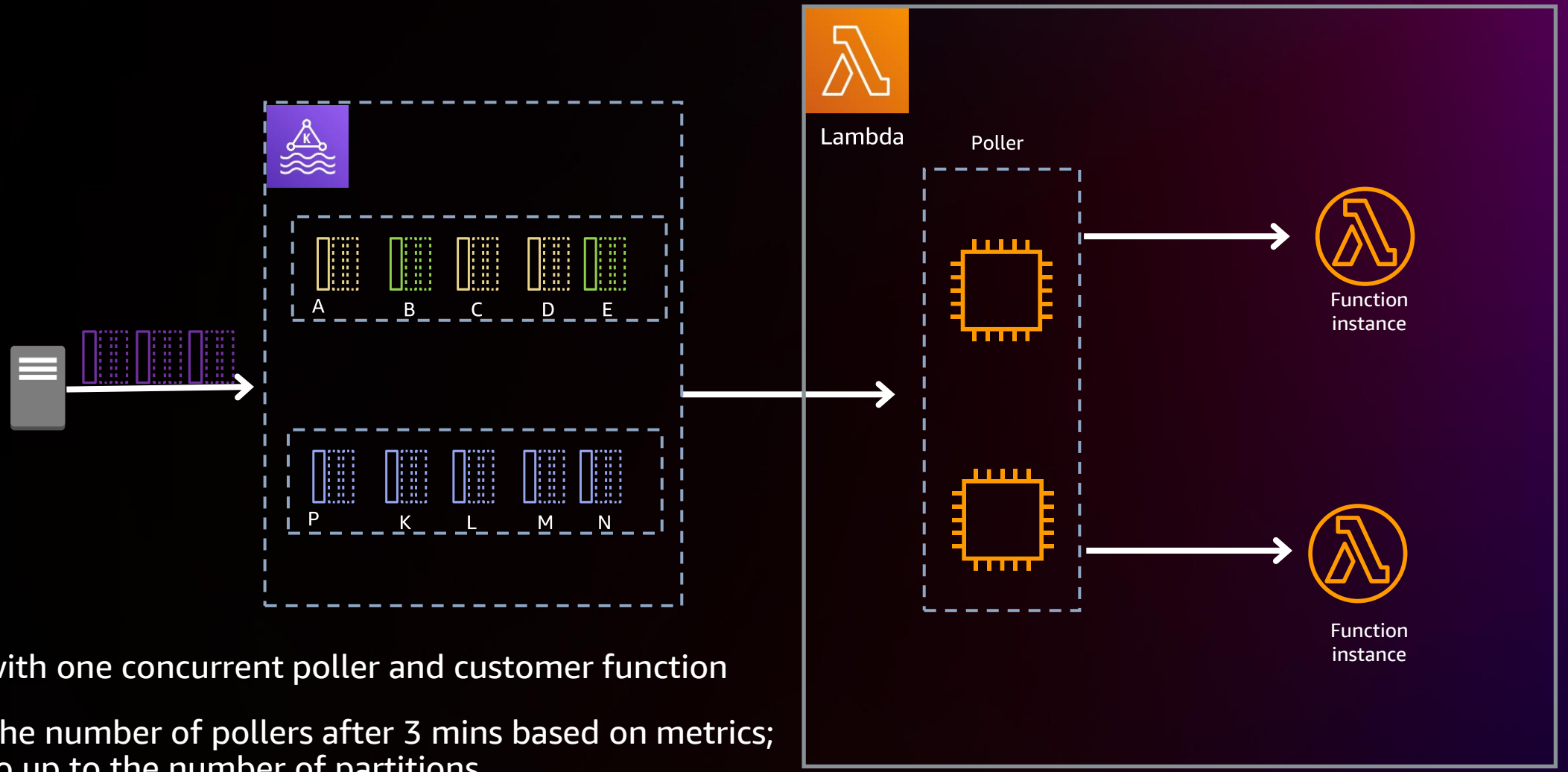
Amazon MSK to Lambda: Poller




```
{   "eventSource": "aws:kafka",
    "eventSourceArn": "arn:aws:kafka:sa-east-1:123456789012:cluster/vpc-2priv-
2pub/751d2973-a626-431c-9d4e-d7975eb44dd7-2",
    "records": {
      "mytopic-0": [
         {
           "topic": "mytopic"
           "partition": "0",
           "offset": 15,
           "timestamp": 1545084650987,
           "timestampType": "CREATE_TIME",
           "value": "SGVsbG8sIHRoaXMgaXMgYSB0ZXN0Lg==",
         }
      ]
    }
}
```


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

Amazon MSK to Lambda: Scaling

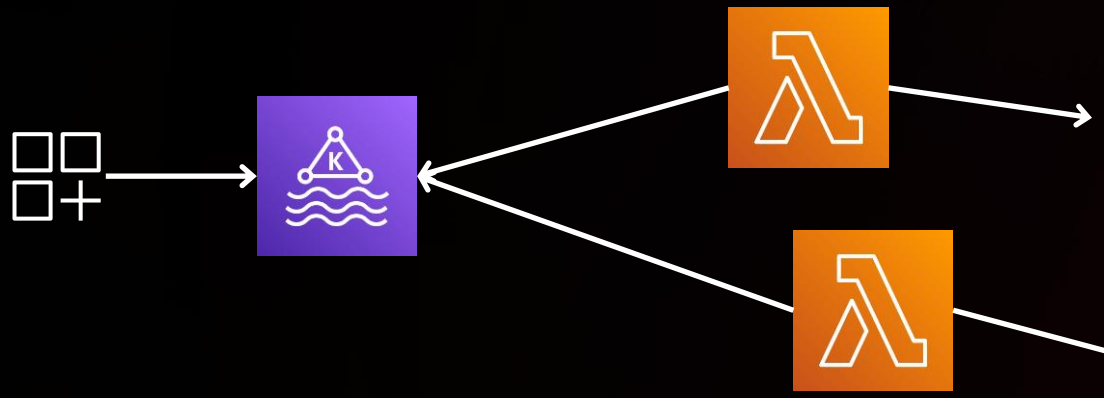


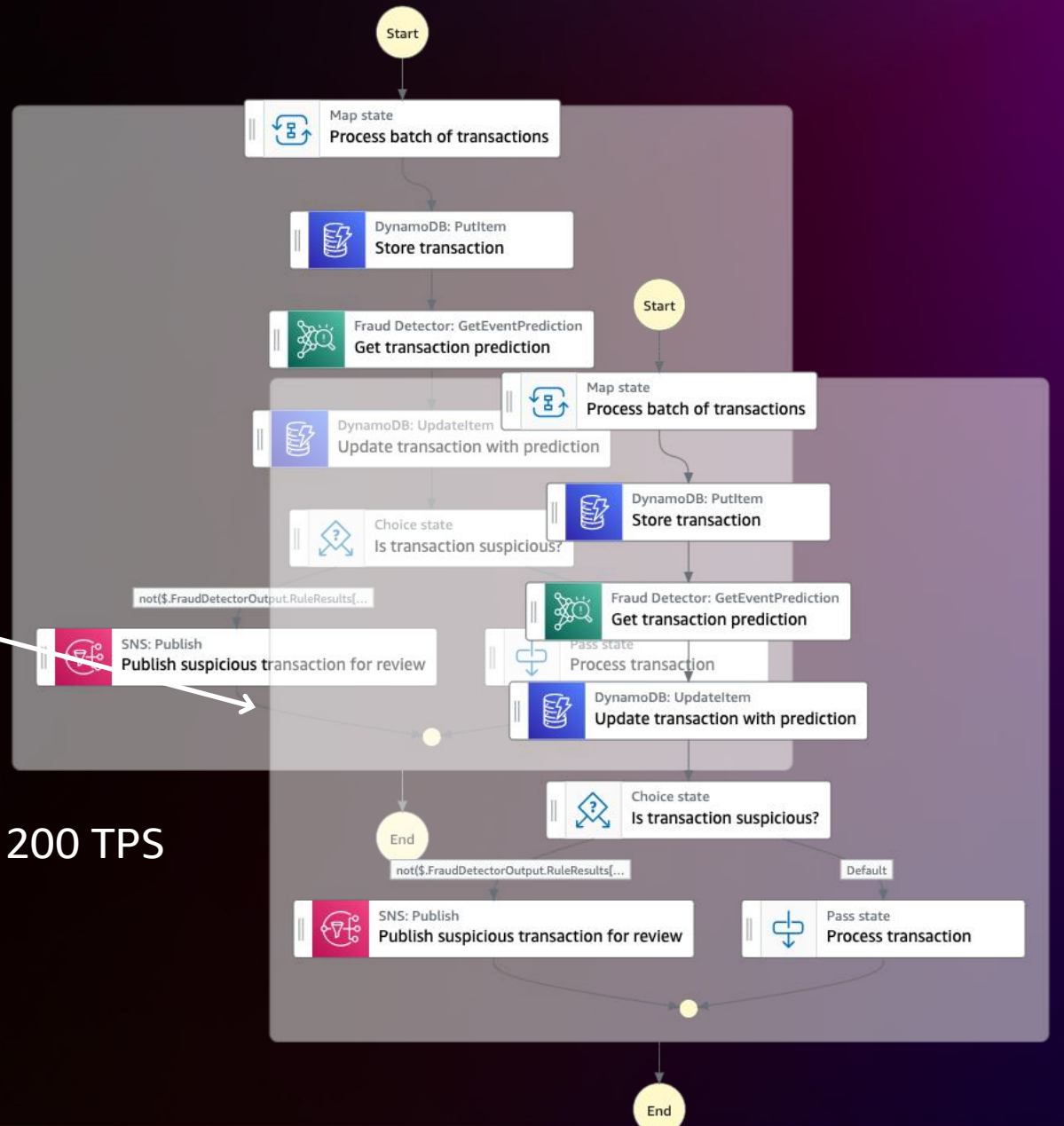
Starts with one concurrent poller and customer function

Scales the number of pollers after 3 mins based on metrics; it can go up to the number of partitions

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

Scaling our fraud detection





Limits to consider:
Step Functions StartExecution API 1300
Amazon Fraud Detector GetEventPrediction API 200 TPS

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

Performance

Cloudwatch metrics to monitor

- Offset lag
- Lambda errors
- Lambda invocations
- Lambda duration
- Lambda concurrency

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

Scalability

MSK serverless

Lambda poller scaling

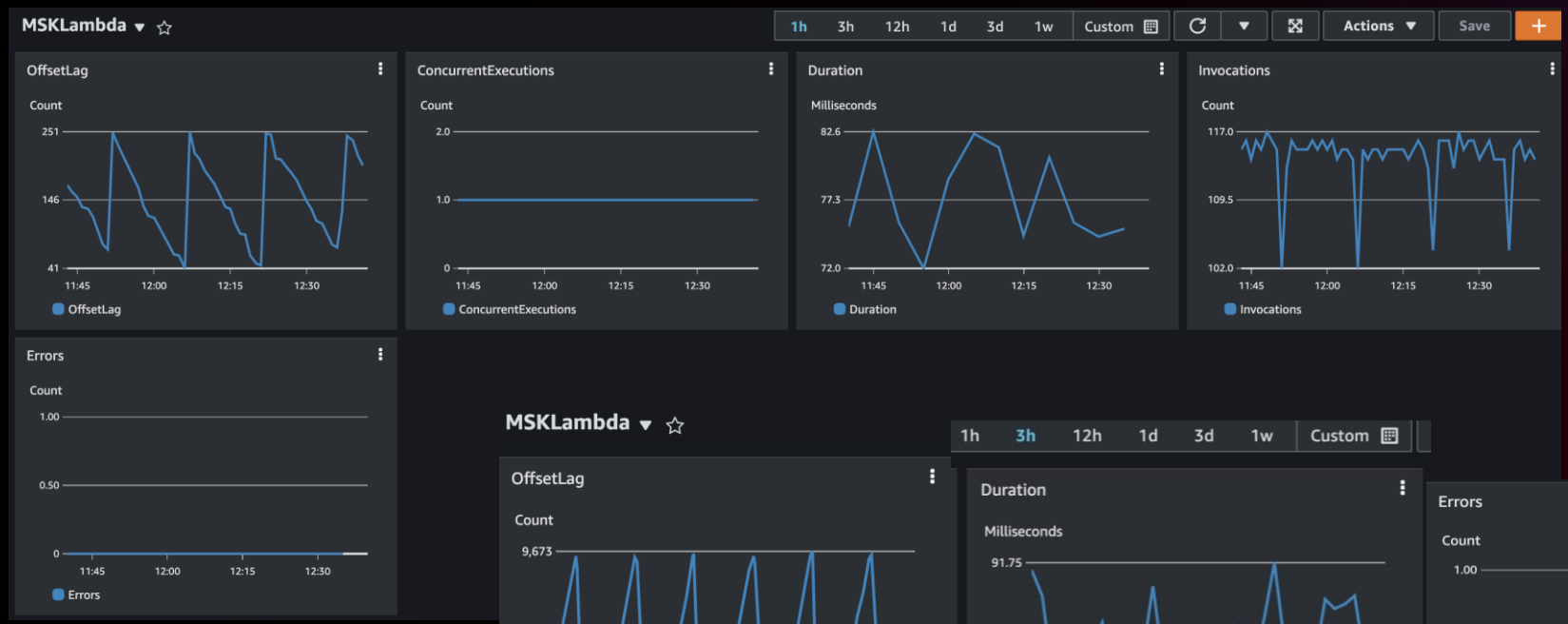
Step Functions scaling

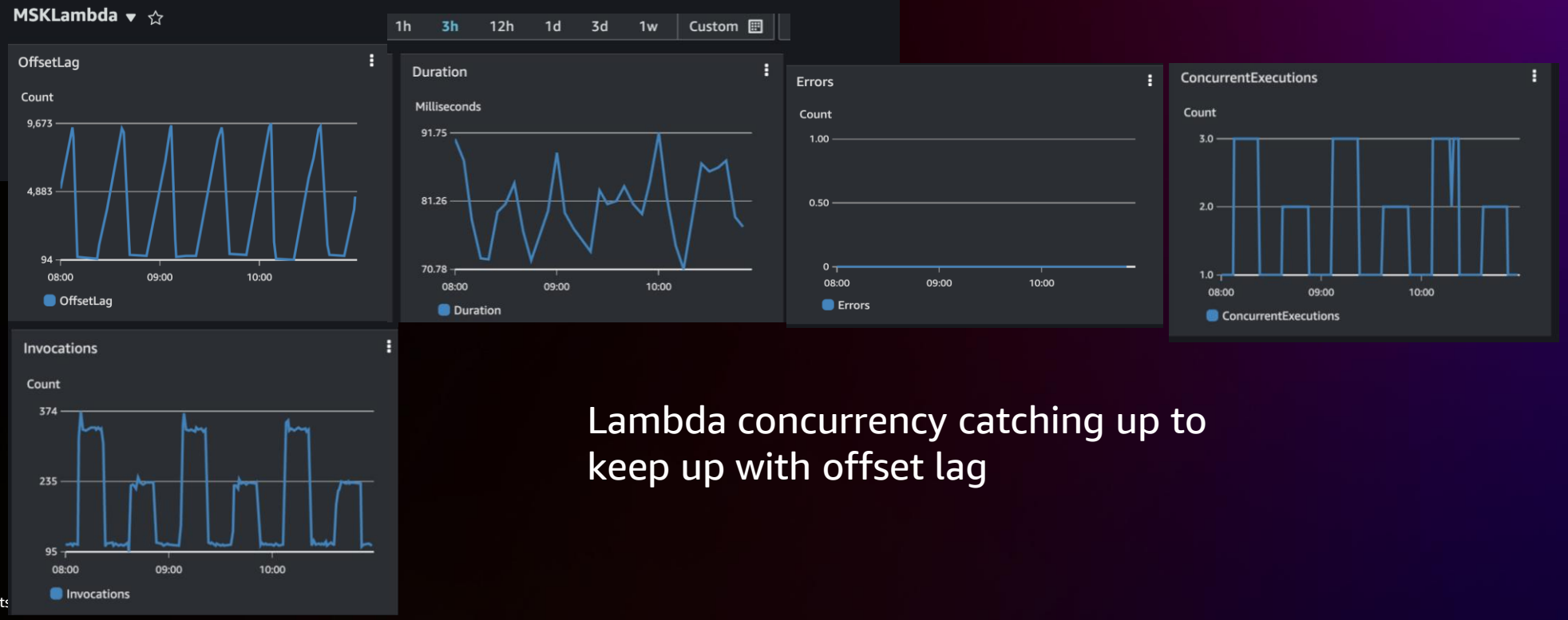
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Troubleshooting

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

Example 1



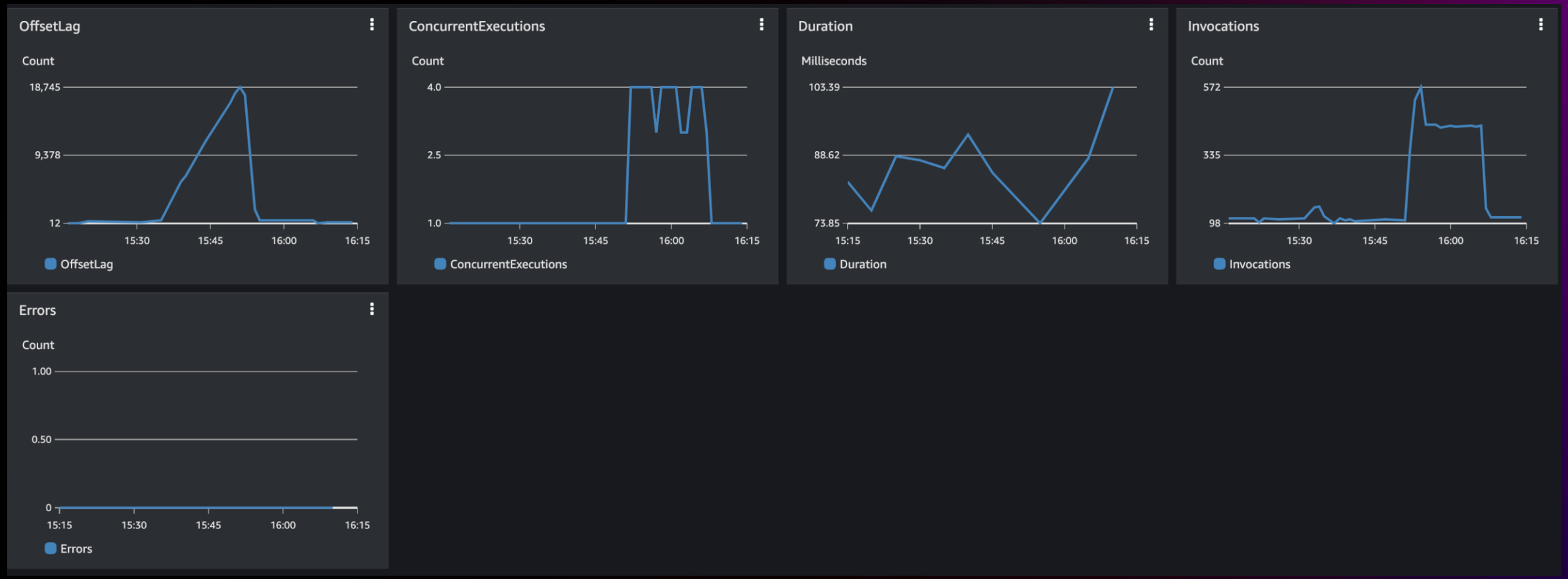


Batch size = 25

Lambda concurrency catching up to keep up with offset lag

© 2022, Amazon Web Services, Inc. or its

Example 2

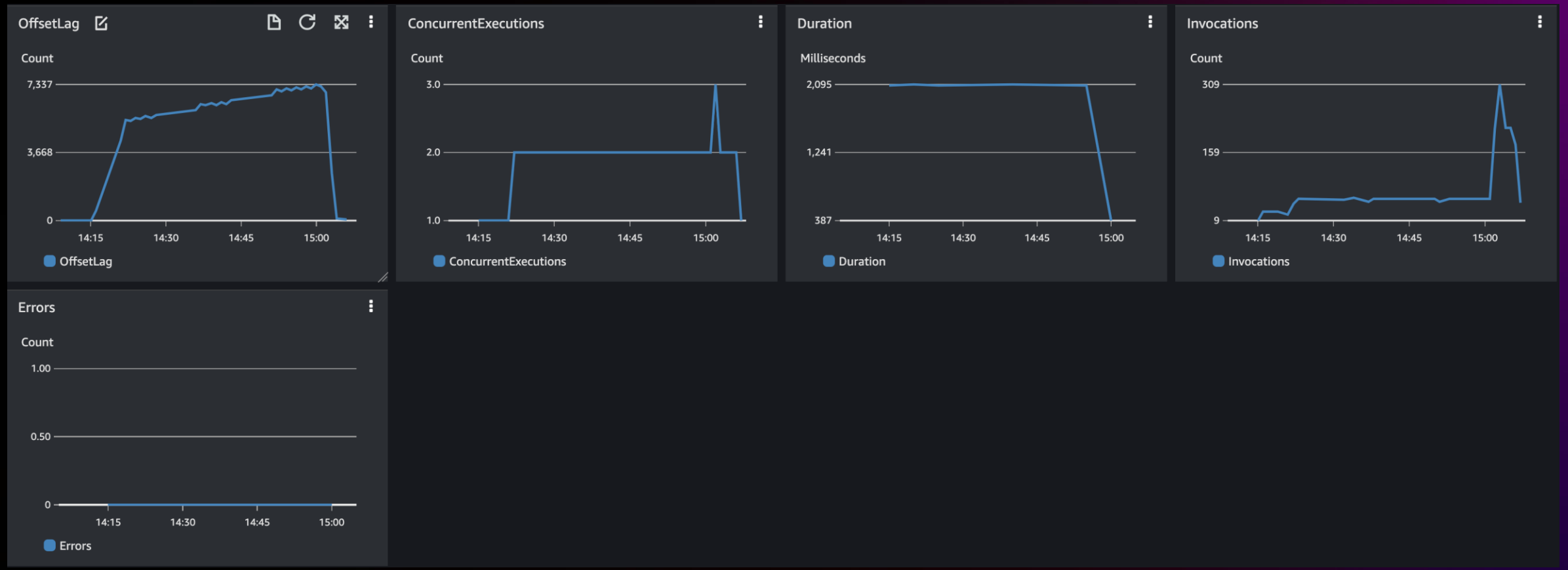


With a traffic spike, a larger batch size helps with keeping offset lag low

aws

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

Example 3



Lambda function duration can influence offset lag

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

Custom consumer group ID

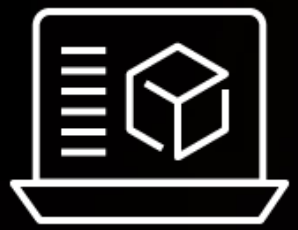
- Any consumer of a Kafka topic uses a consumer croup. In the past, the Lambda event source map created this with a random UUID-ish name.
- This new feature will allow customers to create an event source mapping that uses an existing consumer croup.
- This is important because it enables two important use cases:
  - Connect Lambda to a consumer group configured to start consuming messages at a specific offset or timestamp, something you can't do in the event source mapping configuration.
  - Connect Lambda to a topic that is replicated using MirrorMaker 2.0.

Custom consumer group ID blog

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

Continue your AWS Serverless learning

Learn at your own pace



Expand your serverless skills with our learning plan on **AWS Skill Builder**

Increase your knowledge



Use our **Ramp-Up Guides** to build your serverless knowledge

Earn AWS Serverless badge



Demonstrate your knowledge by achieving **digital badges**

https://s12d.com/serverless-learning

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

Related sessions

SVS316	Tuning AWS Lambda event source mappings – Chalk Talk
ANT318	Build event-based microservices with AWS streaming services – Chalk Talk
BOA325	Building a serverless Apache Kafka data pipeline – Workshop
PRT307	Building real-time serverless data applications with Confluent and AWS (sponsored by Confluent) – Breakout Session

aws

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

Thank you!

Veda Raman
@goofball_veda

Adam Wagner
@oathead

@oathead@hachyderm.io



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

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