When to use a Lambda function? And when not?

Serverless Days Paris 2023

Jerome Van Der Linden (@jeromevdl)

Sr Solutions Architect Builder @ AWS

Who am I ?

Jérôme Van Der Linden

- Solutions Architect Builder @ AWS, Geneva (Switzerland)
- Passionate about Serverless, automation & testing
- Former architect, agile & devops coach, Java developer, Android tech lead
- Husband and father of 3

@jeromevdl

You said AWS Lambda?

You said AWS Lambda ?

Serverless Compute

Run code without provisioning or managing infrastructure. Simply write and upload code as a .zip file or container image.

Automatically respond to code execution requests at any scale, from a dozen events per day to hundreds of thousands per second.

Save costs by paying only for the compute time you use—by per-millisecond—instead of provisioning infrastructure upfront for peak capacity.

You said AWS Lambda ?

Event-driven functions

Event source

Changes in data state

Changes in resource state

Request to endpoints

Function

Node.js
Python
Java
C#
Go
Ruby
Runtime API

Services (anything)

Database

AWS Service

3rd party service

You said AWS Lambda ?

Event-driven functions

Anatomy of a Function

```
export const handler = async function(event, context) {
    console.log('Hello world!');
}
```

- **Handler**: function that will be triggered for each event received

You said AWS Lambda ?

Event-driven functions

Anatomy of a Function

```
export const handler = async function(event, context) {
    console.log('Hello world!');
}
```

- **Handler**: function that will be triggered for each event received
- **Event**: the incoming event that triggered the function (in JSON format)

You said AWS Lambda ?

Event-driven functions

Anatomy of a Function

```
export const handler = async function(event, context) {
    console.log('Hello world!');
}
```

- **Handler**: function that will be triggered for each event received
- **Event**: the incoming event that triggered the function (in JSON format)
- **Context**: informations about the function configuration and invocation

You said AWS Lambda ?

Event-driven functions

Anatomy of a Function

```
export const handler = async function(event, context) {
  console.log('Hello world!');
}
```

- **Handler**: function that will be triggered for each event received
- **Event**: the incoming event that triggered the function (in JSON format)
- **Context**: informations about the function configuration and invocation
- Your code, anything, …

You said AWS Lambda ?

Use-cases

APIs / microservices (REST / GraphQL)

Amazon API Gateway — AWS Lambda

AWS AppSync — AWS Lambda

Orchestration

AWS Step Functions — AWS Lambda

Operations/ Remediation

AWS Config — AWS Lambda

Event-driven architectures

Amazon EventBridge — AWS Lambda

Amazon SNS — AWS Lambda

Amazon SQS — AWS Lambda

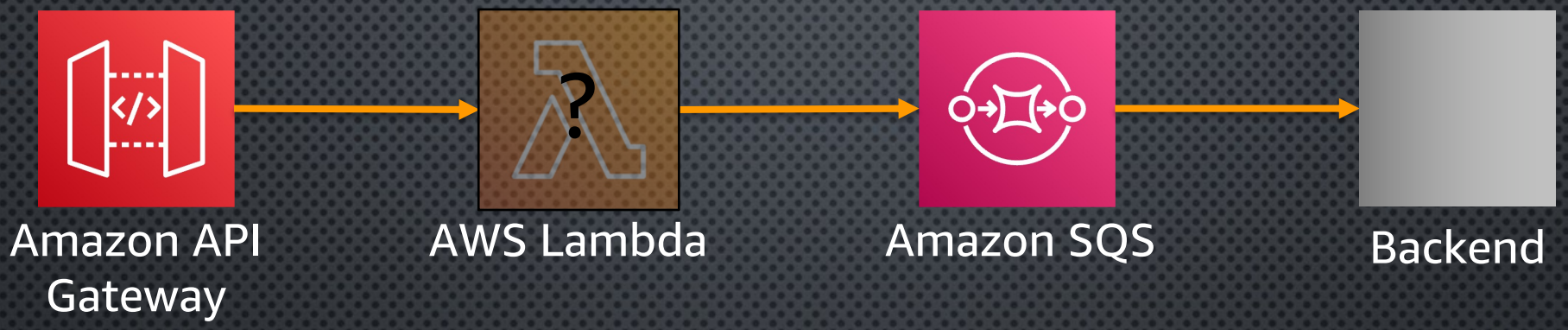
- File processing
- Stream processing
- Data processing (transformation , cleansing, ...)
- Analytics
- IoT backend
- ...

λ or not λ ?
That is the question !

#1 - High-criticality API

High-criticality API

Do we need a Lambda function ?

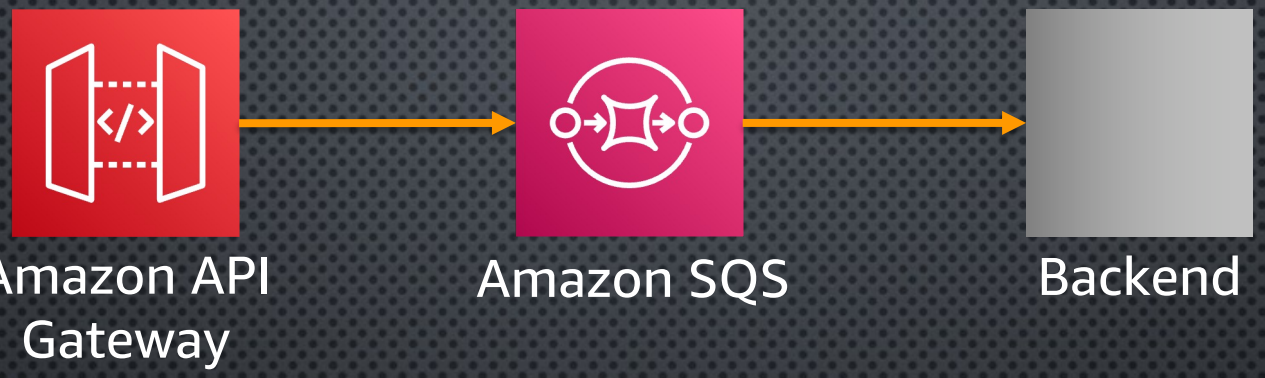


Requirements:

- Users call an API to place orders
- Orders are sent to a queue to be processed asynchronously by a backend
- Incoming orders must not be lost

High-criticality API

NO, Lambda function is not required



Explanation:

- We don't want to loose any order, store them as quickly as possible in the queue
- Having a λ function introduces code and risk of failures (risk of loosing orders)
- This is called the **Storage-first pattern***: store the data before any processing

* https://aws.amazon.com/blogs/compute/building-storage-first-applications-with-http-apis-service-integrations/

High-criticality API

Implementation: leverage API Gateway Direct Integration

REST APIs

- Almost any AWS API
- Using Mapping Template
  - Velocity Template Library (Apache)
- Example:

```
Action=SendMessage&MessageBody=$util.urlEncode("$input.body")
```

HTTP APIs

- EventBridge-PutEvents
- SQS-SendMessage
- SQS-ReceiveMessage
- SQS-DeleteMessage
- SQS-PurgeQueue
- AppConfig-GetConfiguration
- Kinesis-PutRecord
- StepFunctions-StartExecution
- StepFunctions-StartSyncExecution
- StepFunctions-StopExecution

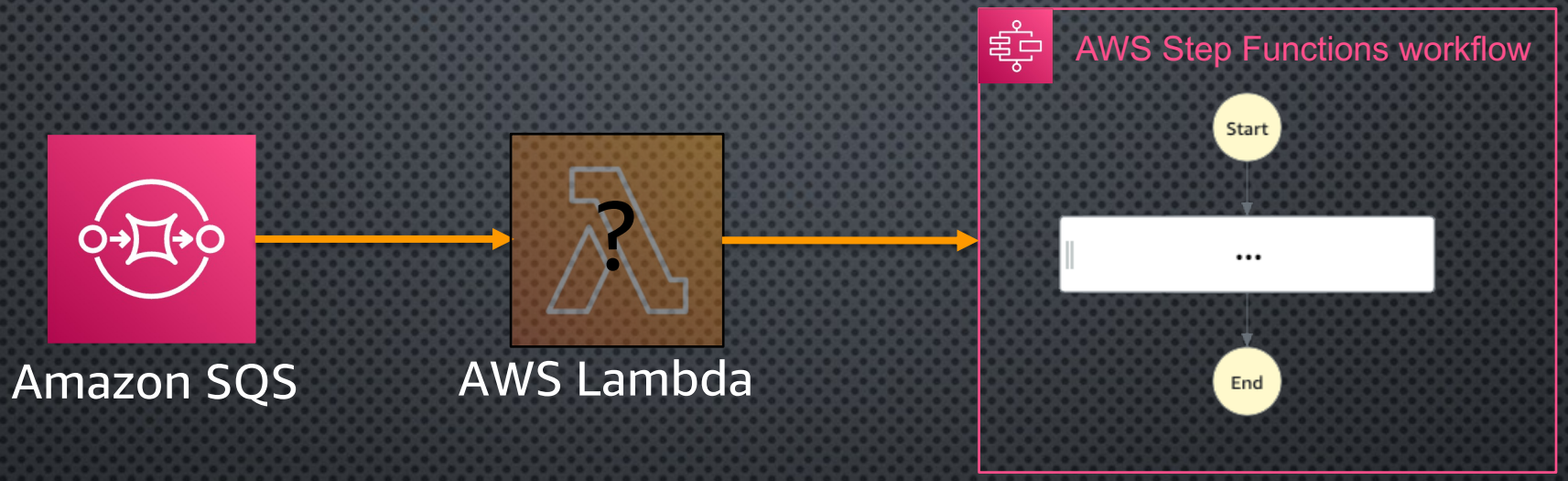
@aws-solutions-constructs/aws-apigateway-sqs

Serverless Land

#2 - Queue & Workflow

Queue & Workflow

Do we need a Lambda function ?

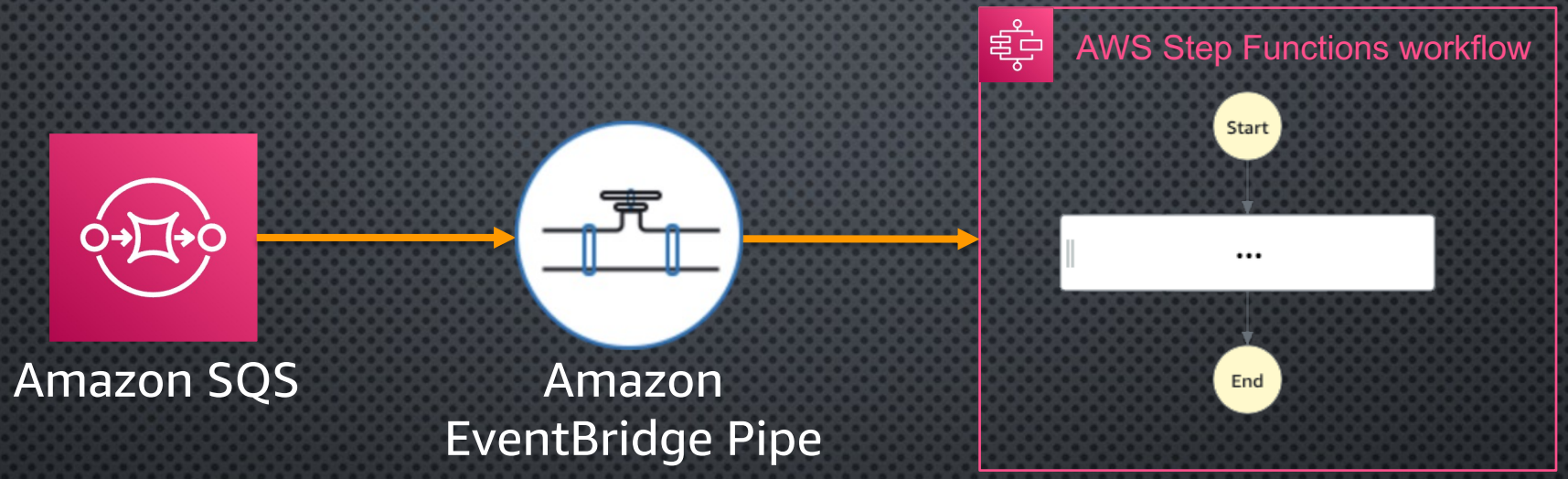


Requirements:

- The SQS Queue receives messages (eg. Orders)
- The backend needs to perform multiple operations on these messages (using Step Functions)

Queue & Workflow

NO, Lambda function is not required

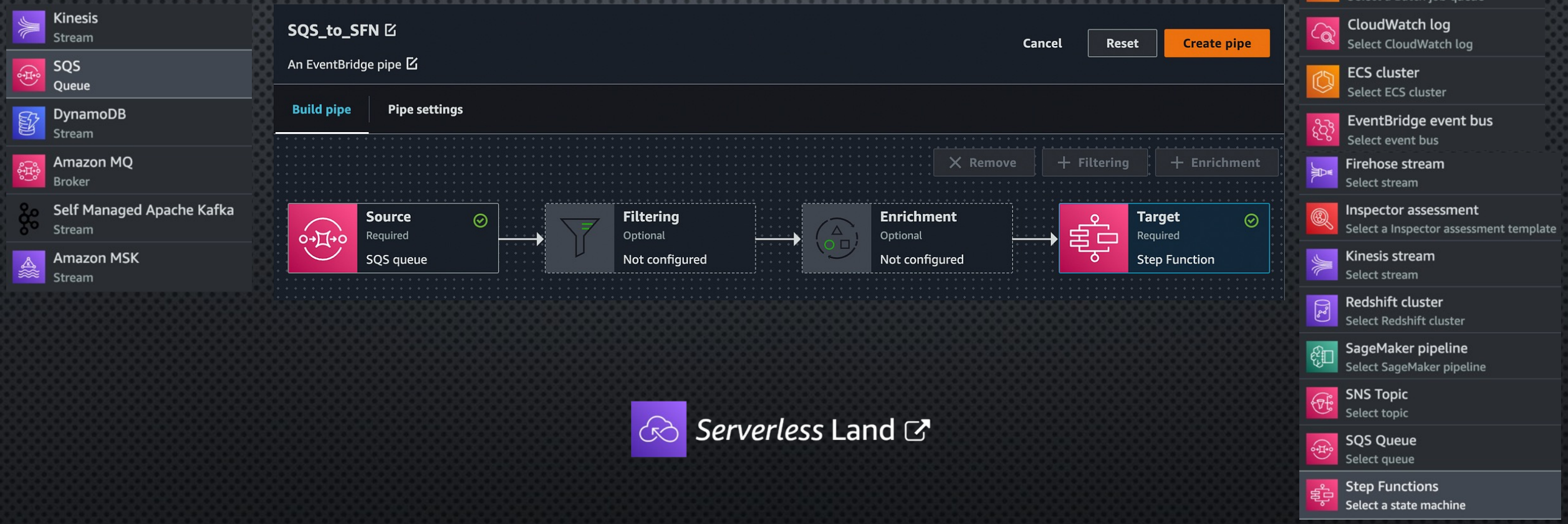


Explanation :

- EventBridge Pipes permit to create point-to-point integration between a source and a target
- No code required
- Use Lambda to transform data, not to transport data

Queue & Workflow

Implementation



Serverless Land

#3 - CQRS

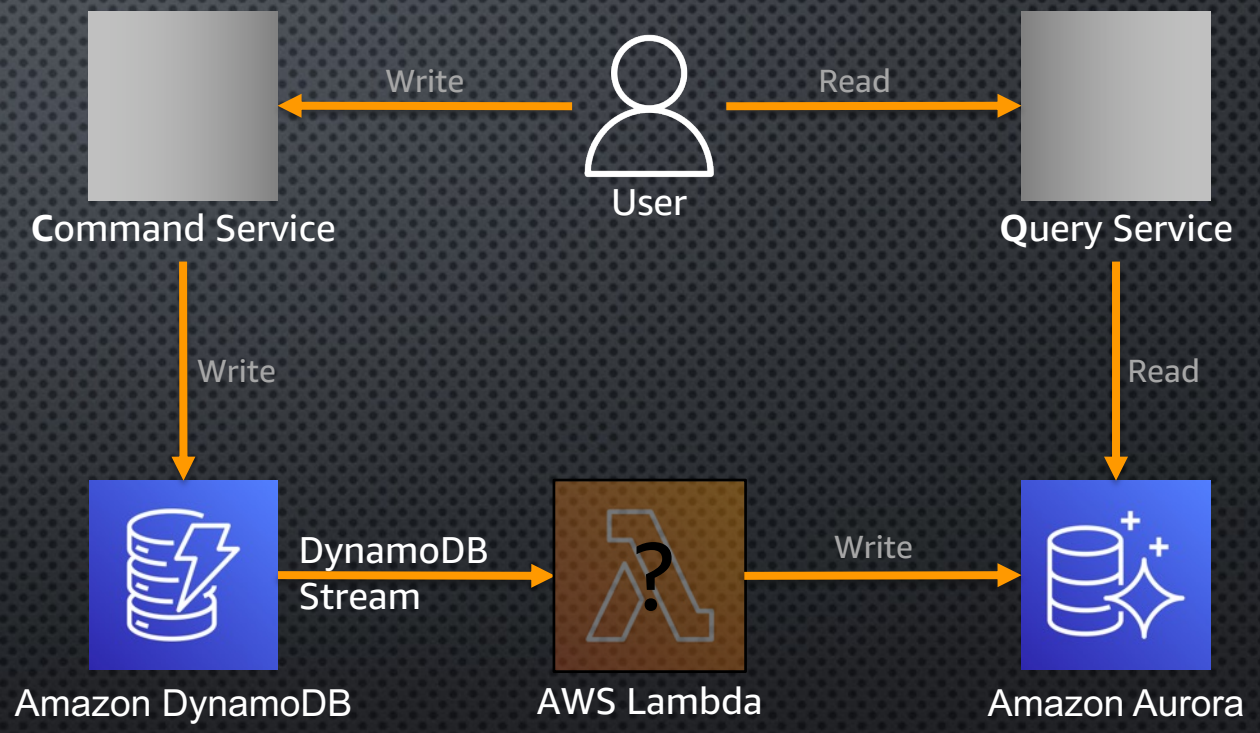
CQRS

Do we need a Lambda function ?

Requirements:

- Different data access patterns (write-intensive / low & structured read)
- Decouple writes from reads to use the best tool for the job

➔ Implement the Command Query Responsibility Segregation pattern

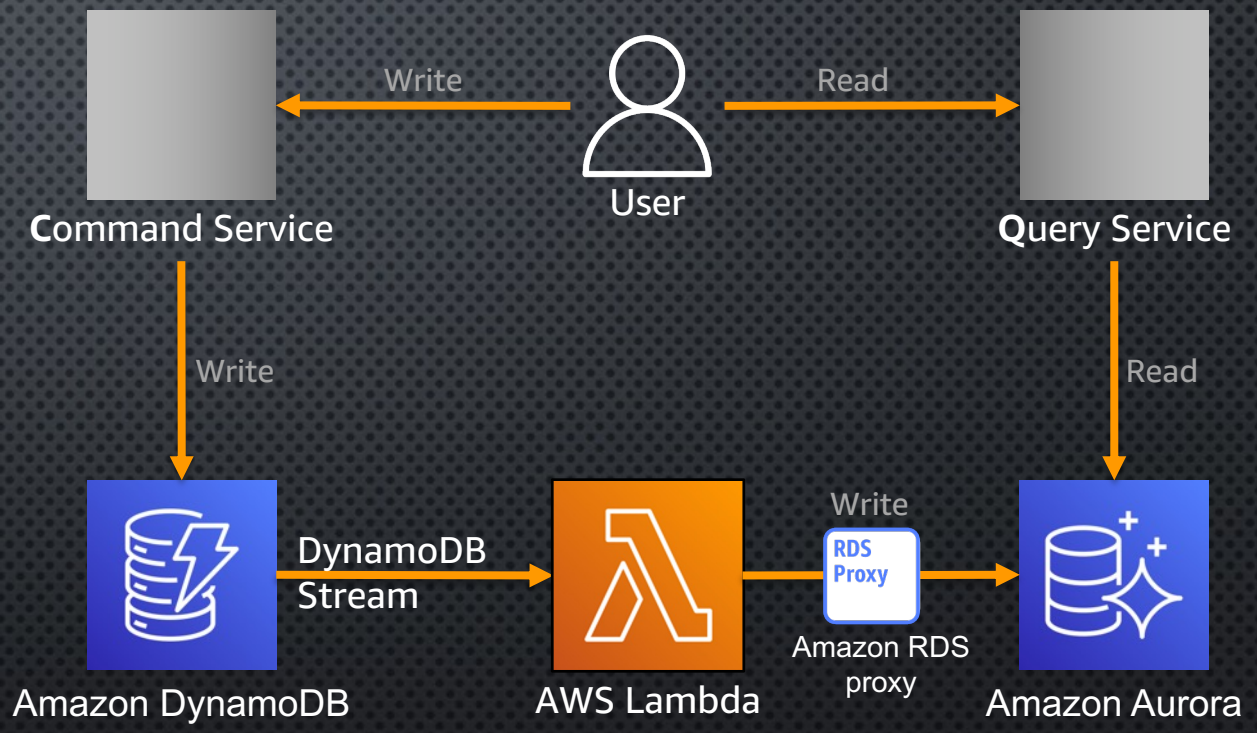


CQRS

YES, Lambda function is required

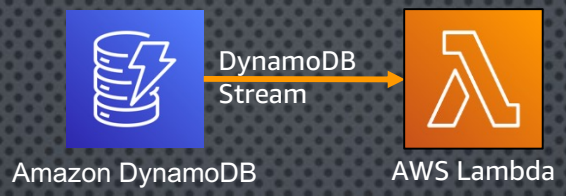
Explanation:

- DynamoDB Streams can only stream to Lambda or Kinesis Data Stream
- EventBridge Pipes do not permit to write to Aurora
- Some compute is required to do the insertion in Aurora
- Using Lambda (with limited reserved concurrency) + RDS proxy also permits to avoid overloading the relational database



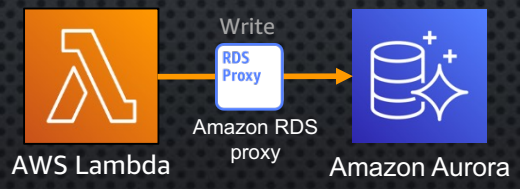
CQRS

Implementation



@aws-solutions-constructs/aws-dynamodbstreams-lambda

Serverless Land

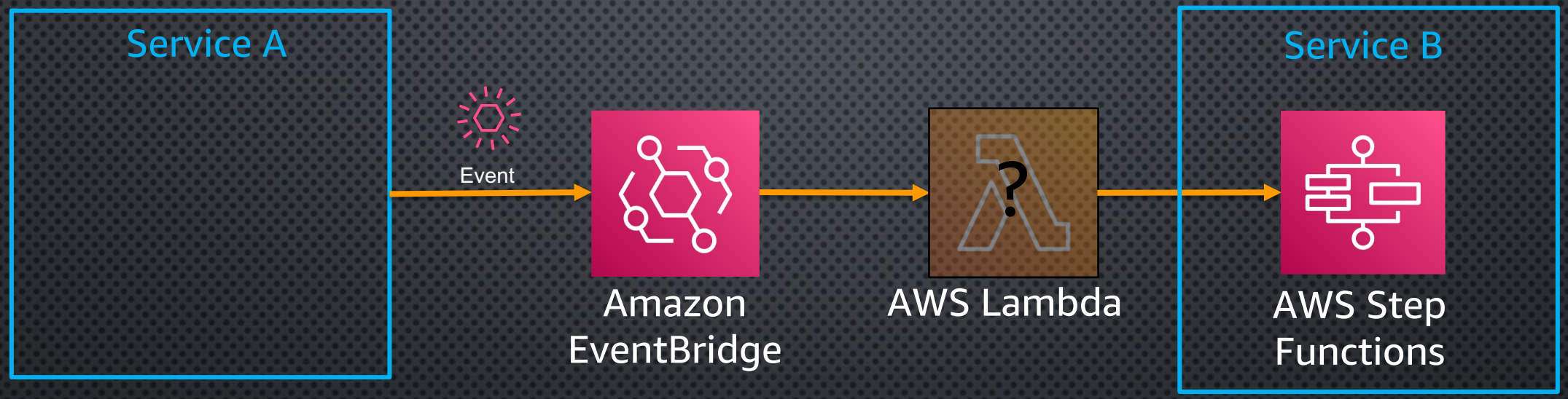


Serverless Land

#4 – Choreography

Choreography

Do we need a Lambda function ?

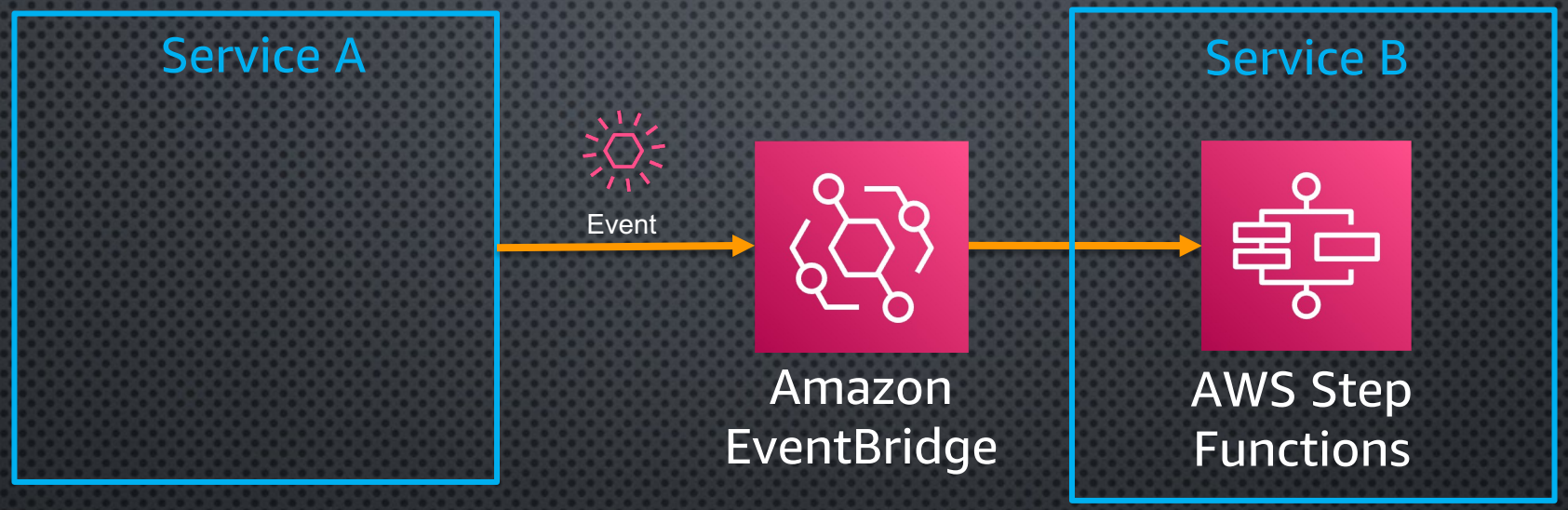


Requirements:

- Keep (µ)services independent and unaware of the others and the overall flow
- Choreography between services for loose coupling and better resilience
  - Use EventBridge and events

Choreography

NO, Lambda function is not required



Explanation:

- EventBridge provides direct integration with Step Functions
- And many others: https://docs.aws.amazon.com/eventbridge/latest/userguide/eb-targets.html

Choreography

Implementation

Target 1

Target types

Select an EventBridge event bus, EventBridge API destination (SaaS partner), or another AWS service as a target.

- ○ EventBridge event bus
- ○ EventBridge API destination
- ◉ AWS service

Select a target | Info

Select target(s) to invoke when an event matches your event pattern or when schedule is triggered (limit of 5 targets per rule)

Step Functions state machine

State machine

Select state machine

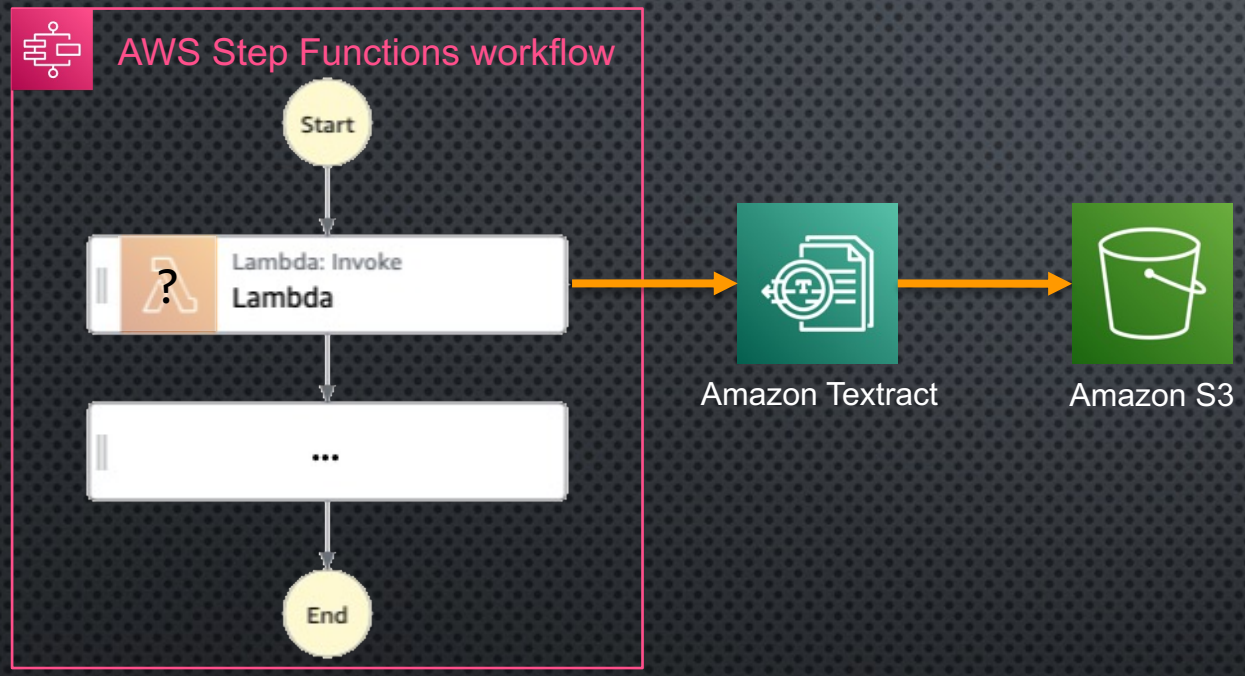
@aws-solutions-constructs/aws-eventbridge-stepfunctions

Serverless Land

#5 – AWS service orchestration

AWS service orchestration

Do we need a Lambda function ?

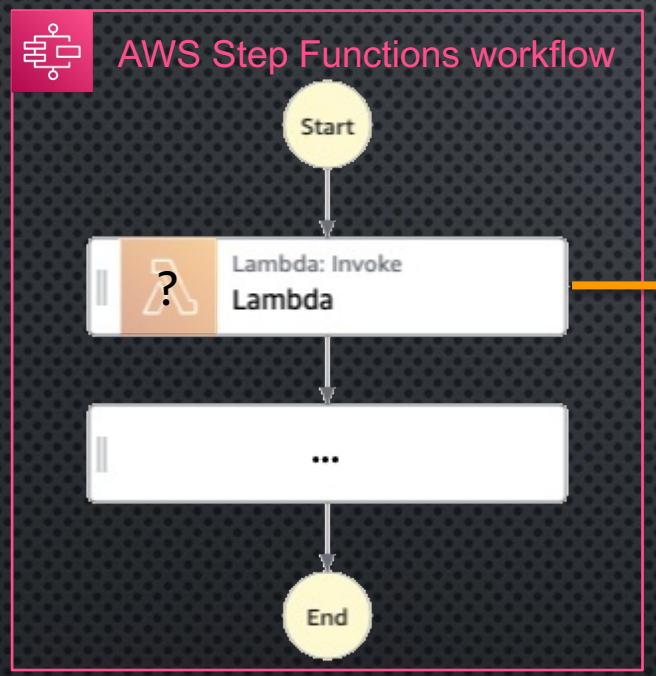


Requirements:

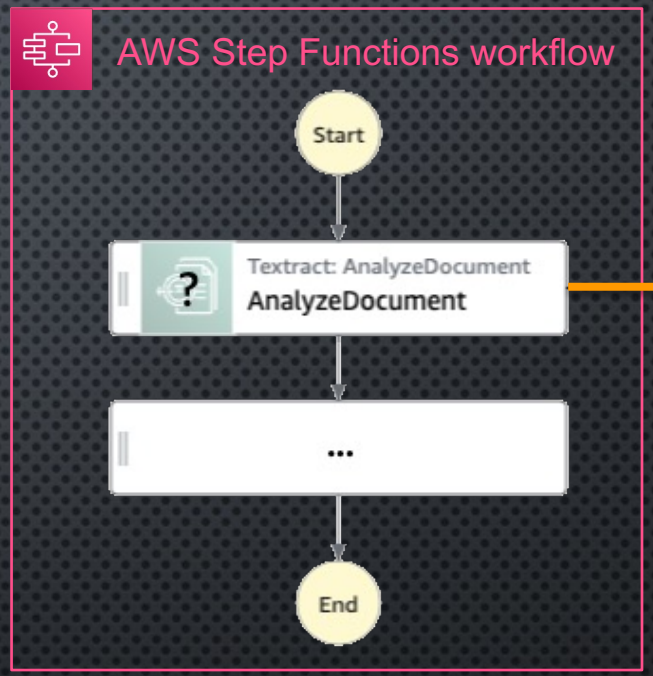
- Orchestrate different operations within a Step Functions workflow
  - Analyze PDF documents (using Textract)
  - Extract meaningful information and perform others operations on it

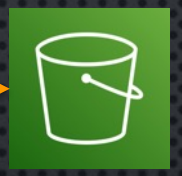
AWS service orchestration

Do we need a Lambda function ?



OR



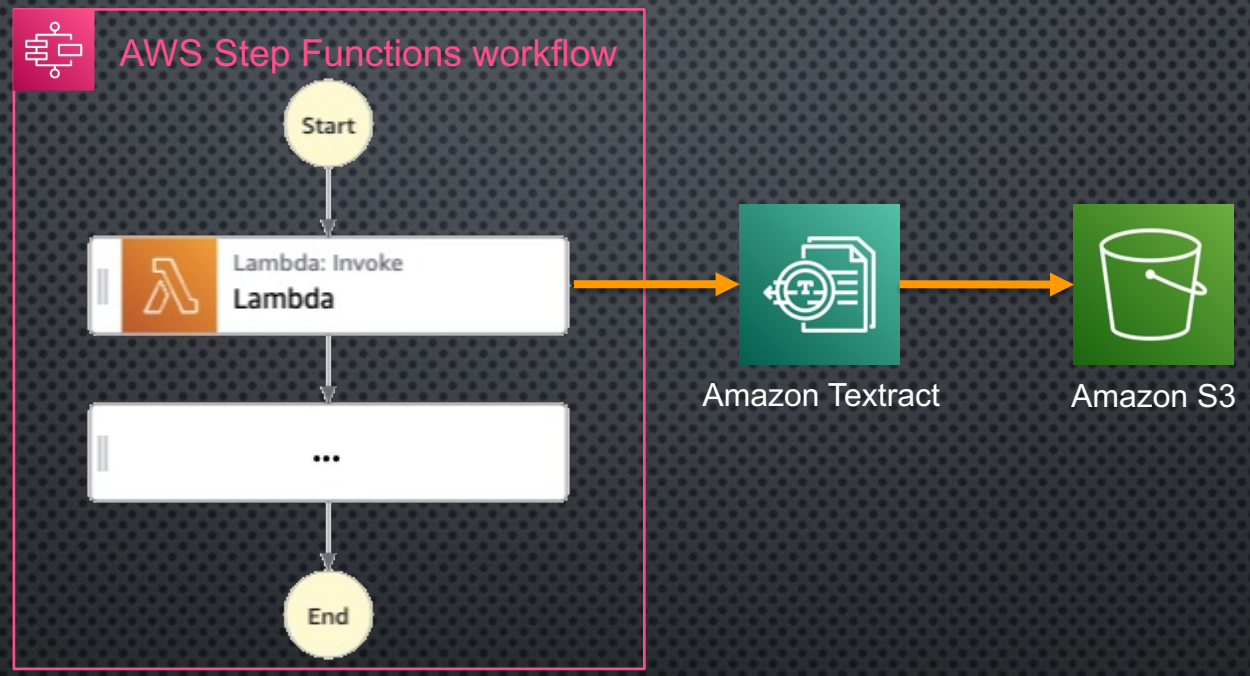


Requirements:

- Orchestrate different operations within a Step Functions workflow
  - Analyze PDF documents (using Textract)
  - Extract meaningful information and perform others operations on it

AWS service Orchestration

Most probably NO, but in this case YES



Explanation:

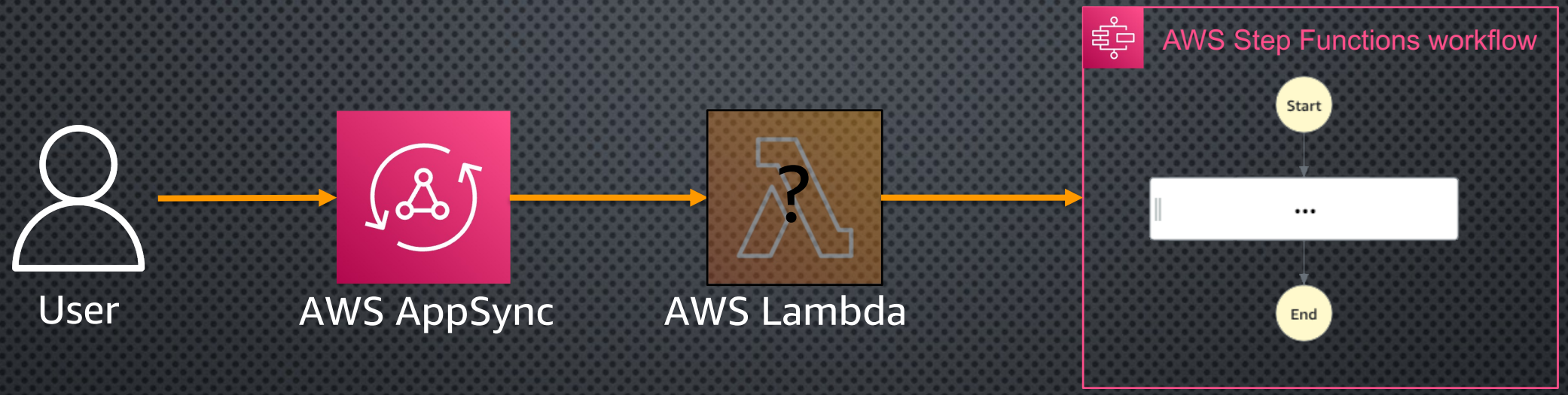
- Step Functions provides direct integration with almost all AWS APIs*
- Step Functions provides more & more Intrinsic functions to manipulate JSON and arrays
- BUT Amazon Textract responses are very verbose and require a parser and therefore code:
  - https://github.com/aws-samples/amazon-textract-response-parser

*: https://docs.aws.amazon.com/step-functions/latest/dg/supported-services-awssdk.html

#6 – GraphQL API triggering a workflow

GraphQL API triggering a workflow

Do we need a Lambda function ?



Requirements:

- Instead of a REST API (using API Gateway), we use a GraphQL API (using AppSync)
- We don't want to use a Pipeline Resolver but leverage a Step Functions workflow
- Execution of the workflow must be synchronous

GraphQL API triggering a workflow

NO, Lambda function is not required

Explanation:

- Appsync does not integrate natively with Step Functions
- Appsync provides HTTP Resolvers to perform HTTP calls
- All AWS APIs are backed by an HTTP API* and callable by HTTP Resolvers

*: https://docs.aws.amazon.com/general/latest/gr/aws-service-information.html

GraphQL API triggering a workflow

Implementation

Create new Data Source

Data source name

StepFunctionsEndpoint

A name starts with a letter and contains only numbers, lette

Data source type

Select the type of your data source.

HTTP endpoint

HTTP endpoint

Enter the base url of the HTTP endpoint.

https://sync-states.eu-west-1.amazonaws.com

Configure the request mapping template. Info

Translate a GraphQL query into a format specific to your data source.

```
{
   "version": "2018-05-29",
   "method": "POST",
   "resourcePath": "/",
   "params": {
       "headers": {
           "content-type": "application/x-amz-json-1.0",
           "x-amz-target":"AWSStepFunctions.StartSyncExecution"
        },
        "body": {
            "stateMachineArn": "arn:aws:states:eu-west1:123456789012:stateMachine:my-state-machine",
            "input": "{ \"details\": \"$context.args.input\"}"
        }
    }
}
```

Serverless Land

https://aws.amazon.com/blogs/mobile/invoking-aws-step-functions-short-and-long-running-workflows-from-aws-appsync/

#7 – Uploading big files to Amazon S3

Uploading big files to Amazon S3

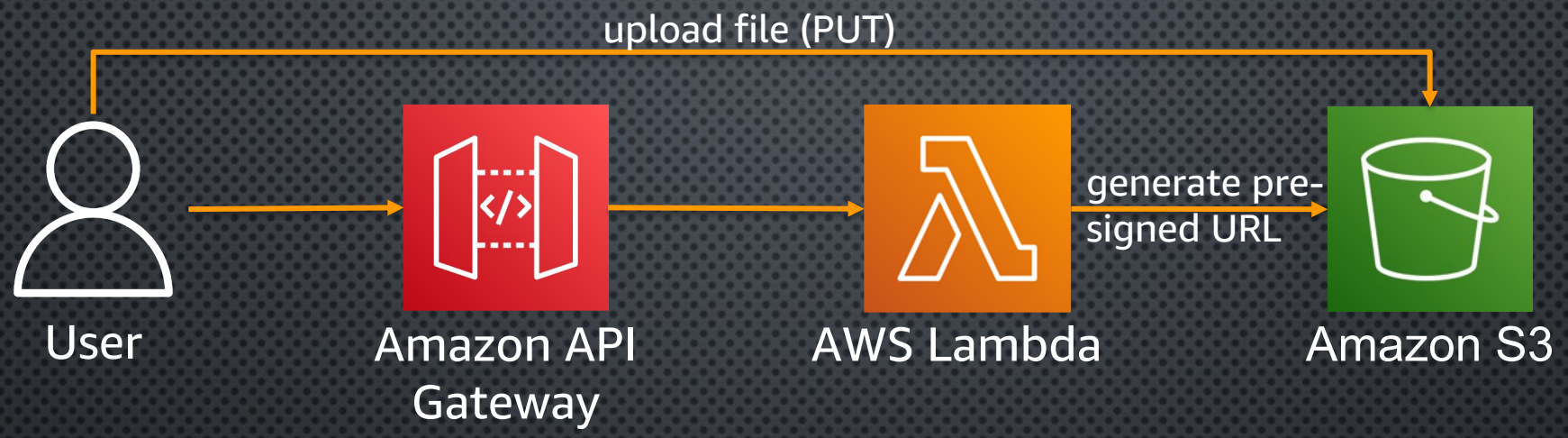
Do we need a Lambda function ?

Requirements:

- Users need to upload big files (> 10MB)
- Files need to be stored as objects in S3

Uploading big files to Amazon S3

YES, Lambda function is required



Explanation:

- API Gateway does not support payloads > 10MB (6MB for Lambda)
  ➔ We need to generate a S3 pre-signed URL the user will use for the upload

- Despite its multiple direct integrations, API Gateway cannot call the pre-signing API
  - Not part of the CLI `s3api` command, not part of the SDK @aws-sdk/client-s3
  - Part of the CLI `s3` command, part of the SDK `@aws-sdk/s3-request-presigner`

Uploading big files to Amazon S3

Implementation

Serverless Land

```
import { S3Client, PutObjectCommand } from '@aws-sdk/client-s3';
import { getSignedUrl } from '@aws-sdk/s3-request-presigner';
import { extension as getExtension } from 'es-mime-types';

const client = new S3Client({
  region: process.env.AWS_REGION,
});

exports.handler = async (event: any) => {
  const uploadURL = await getUploadURL(event);

  return {
    statusCode: 200,
    body: JSON.stringify(uploadURL),
  };
};

const getUploadURL = async function(event: any) {

  const apiRequestId = event.requestContext.requestId;
  const contentType = event.queryStringParameters.contentType;
  const extension = getExtension(contentType);
  const s3Key = `${apiRequestId}.${extension}`;

  // Get signed URL from S3
  const putObjectParams = {
    Bucket: process.env.UPLOAD_BUCKET,
    Key: s3Key,
    ContentType: contentType,
  };
  const command = new PutObjectCommand(putObjectParams);

  const signedUrl = await getSignedUrl(client, command, {
    expiresIn: parseInt(process.env.URL_EXPIRATION_SECONDS || '300')
  });

  return {
    uploadURL: signedUrl,
    key: s3Key,
  };
};
```

+ allow `s3:putObject` to the function

Conclusion

Conclusion

- When available, privilegiate native direct integration between services
  - API Gateway / EventBridge / Step Functions
- If not available, look for HTTP integrations (AWS services are just APIs)
  - AppSync HTTP Resolvers / (EventBridge API destinations)
- If not available, too complex to implement / requires code, use λ
  - VTL mapping templates are not really 'business-proof'
- Use Lambda to transform data, not to transport data

aws

Thank you!

Jerome Van Der Linden
@jeromevdl