AWS Solutions

AWS Solutions Constructs

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Solutions Constructs: AWS Solutions

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

AWS Solutions Constructs

Publication date: **April 2024** (*Document Revisions*)

What is AWS Solutions Constructs?

AWS Solutions Constructs (Constructs) is an open-source extension of the AWS Cloud Development Kit (AWS CDK) that provides multi-service, well-architected patterns for quickly defining solutions in code to create predictable and repeatable infrastructure. The goal is to accelerate the experience for developers to build solutions of any size using pattern-based definitions for their architecture.

Use the AWS Solutions Constructs to define your solutions in a familiar programming language. The AWS Solutions Constructs supports TypeScript, JavaScript, Python, and Java at this time.

To browse the full catalog of AWS Solutions Constructs patterns, click here.

Why use AWS Solutions Constructs?

With the rate of innovation of cloud providers, knowing and understanding best practices and ensuring they are implemented correctly across your solution can be daunting. Constructs allows you to combine pre-built, well-architected patterns and use cases that perform common actions using cloud services in a scalable and secure manner. Because Constructs provides a library for modern programming languages, you can apply existing development skills and familiar tools to the task of building well-architected cloud infrastructure for your solutions.

Other advantages of AWS Solutions Constructs include:

- It is built upon the AWS Cloud Development Kit (AWS CDK) open source software development framework.
- Use logic (if statements, for-loops, etc.) when defining your solution infrastructure.
- Use object-oriented techniques to create a model of your system.
- Define high level abstractions, share them, and publish them to your team, company, or community.
- Organize your solutions into logical modules.
- Share and reuse your solution as a library.
- Test your infrastructure code using industry-standard protocols.
- Use your existing code review workflow.

The aim of AWS Solutions Constructs is to reduce the complexity and glue logic required when integrating common well-architected patterns to achieve your solution goals on AWS.

Getting Started with AWS Solutions Constructs

This topic describes how to install and configure AWS Cloud Development Kit (AWS CDK), AWS Solutions Constructs, and create your first AWS CDK app using AWS Solutions Constructs patterns.

> **Tip**
>
> Want to dig deeper? Try the CDK Workshop for a more in-depth tour of a real-world project.

> **Tip**
>
> For more information about getting started with the AWS Cloud Development Kit (AWS CDK), refer to the AWS CDK Developer Guide.

Prerequisites

AWS Solutions Constructs is built upon the AWS CDK, so you need to install Node.js (>= 10.3.0), even those working in languages other than TypeScript or JavaScript. This is because the AWS CDK and AWS Solutions Constructs are developed in TypeScript and run on Node.js. The bindings for other supported languages use this backend and toolset.

You must provide your credentials and an AWS Region to use the AWS CDK CLI, as described in Specifying Your Credentials and Region.

Other prerequisites depend on your development language, as follows.

Language	Prerequisites
P	Python >= 3.6
Ty t	TypeScript >= 2.7

Language	Prerequisites
Ja	Java >= 1.8

Installing the AWS CDK

To install and configure the AWS CDK, please refer to the AWS CDK Developer Guide - Installing the AWS CDK.

Working with AWS Solutions Constructs

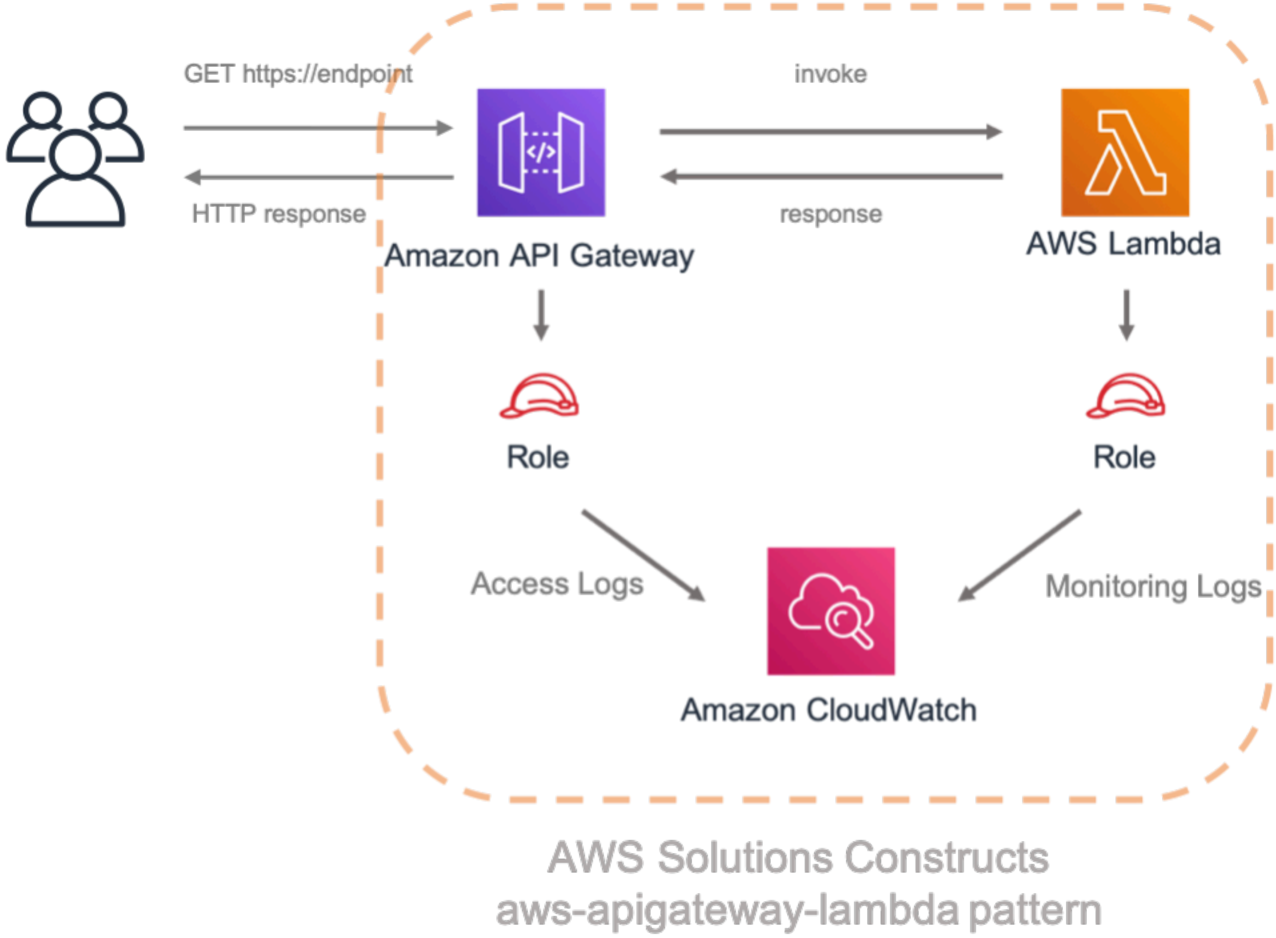
The typical workflow for creating a new app when working with AWS Solutions Constructs follows the same approach as the AWS CDK.

1. Create the app directory.
2. Initialize the app.
3. Add the AWS Solutions Constructs pattern dependencies.
4. Add additional code to the app.
5. Compile the app, if necessary.
6. Deploy the resources defined in the app.
7. Test the app.

If there are any issues, loop through modify, compile (if necessary), deploy, and test again.

Walkthrough - Part 1

This tutorial walks you through how to create and deploy a simple "Hello Constructs" AWS CDK app that uses a pattern from AWS Solutions Constructs, from initializing the project to deploying the resulting AWS CloudFormation template. The Hello Constructs app will create the following simple solution:



Hello Constructs

Let's get started building our first AWS CDK App using AWS Solutions Constructs.

> **Note**
>
> This is a sample modification of `Hello CDK!` from the [CDK Workshop](). If this is your first time using the AWS CDK, we recommend starting with this workshop for a hands-on walkthrough and how to leverage the CDK in building a real-world project.

Creating the App Directory and Initializing the AWS CDK

If you have not installed the AWS CDK yet, do so following the instructions [here](). Confirm you are using version 2.0 or higher of the AWS CDK by running this command.

```
cdk --version
```

Create a directory for your CDK app, and then create a AWS CDK app in that directory.

TypeScript

```
mkdir hello-constructs
cd hello-constructs
cdk init --language typescript
```

Python

```
mkdir hello-constructs
cd hello-constructs
cdk init --language python
```

Java

```
mkdir hello-constructs
cd hello-constructs
cdk init --language java
```

Tip

Now's a good time to open the project in your favorite IDE and explore. To learn more about the project structure, select the appropriate link:

- TypeScript
- Python
- Java

Install project base dependencies

Install the projects base dependencies.

TypeScript

```
npm install
```

Python

```
source .venv/bin/activate
pip install -r requirements.txt
```

Java

```
mvn install
```

Build and run the app and confirm that it creates an empty stack.

TypeScript

```
npm run build
cdk synth
```

Python

```
cdk synth
```

Java

```
mvn package
cdk synth
```

You should see a stack like the following, where `CDK-VERSION` is the version of the CDK. (Your output may differ slightly from what's shown here – the metadata and parameters are omitted from this listing.)

TypeScript

```
Resources:
  CDKMetadata:
    Type: AWS::CDK::Metadata
    Properties:
          Modules: aws-cdk=CDK-VERSION,@aws-cdk/core=VERSION_NUMBER,@aws-cdk/cx-
api=VERSION_NUMBER,jsii-runtime=node.js/VERSION_NUMBER
```

Python

```
Resources:
  CDKMetadata:
    Type: AWS::CDK::Metadata
    Properties:
          Modules: aws-cdk=CDK-VERSION,@aws-cdk/core=VERSION_NUMBER,@aws-cdk/cx-
api=VERSION_NUMBER,jsii-runtime=Python/VERSION_NUMBER
```

Java

```
Resources:
  CDKMetadata:
    Type: AWS::CDK::Metadata
    Properties:
          Modules: aws-cdk=CDK-VERSION,@aws-cdk/core=VERSION_NUMBER,@aws-cdk/cx-
api=VERSION_NUMBER,jsii-runtime=Java/VERSION_NUMBER
```

Lambda handler code

We'll start with the AWS Lambda handler code.

Create a directory `lambda` in the root of your project tree.

TypeScript

Add a file called `lambda/hello.js` with the following contents:

```
exports.handler = async function(event) {
  console.log("request:", JSON.stringify(event, null, 2));
  return {
    statusCode: 200,
    headers: { "Content-Type": "text/plain" },
    body: `Hello, AWS Solutions Constructs! You've hit ${event.path}\n`
  };
};
```

Python

Add a file called `lambda/hello.py` with the following contents:

```
import json

def handler(event, context):
    print('request: {}'.format(json.dumps(event)))
    return {
        'statusCode': 200,
        'headers': {
            'Content-Type': 'text/plain'
        },
        'body': 'Hello, AWS Solutions Constructs! You have hit
 {}\n'.format(event['path'])
    }
```

Java

Add a file called `lambda/hello.js` with the following contents:

```
exports.handler = async function(event) {
  console.log("request:", JSON.stringify(event, null, 2));
  return {
    statusCode: 200,
    headers: { "Content-Type": "text/plain" },
```

```
    body: `Hello, AWS Solutions Constructs! You've hit ${event.path}\n`
  };
};
```

This is a simple Lambda function which returns the text "Hello, Constructs! You've hit [url path]". The function's output also includes the HTTP status code and HTTP headers. These are used by API Gateway to formulate the HTTP response to the user. It's in this function where you would insert logic for your application.

For more information on writing Lambda functions in your language of choice, refer to the AWS Lambda documentation .

Install the AWS Solutions Constructs dependencies

The AWS Solutions Constructs is shipped with an extensive library of constructs. The library is divided into modules, one for each well-architected pattern. For example, if you want to define an Amazon API Gateway Rest API that invokes an AWS Lambda function, we will need to use the `aws-apigateway-lambda` pattern library.

The AWS Lambda and Amazon API Gateway modules are installed with the AWS CDK.

Install the AWS Solutions Constructs `aws-apigateway-lambda` module and all its dependencies into our project:

TypeScript

```
npm install -s @aws-solutions-constructs/aws-apigateway-lambda
```

Python

```
pip install aws_solutions_constructs.aws_apigateway_lambda
```

Java

Edit the `pom.xml` file with the following information:

> **Note**
>
> 2.5.0 was the latest version of AWS Solutions Constructs at this writing. To find the latest version of AWS Solutions Constructs (and any other dependencies), use the command `mvn versions:display-dependency-updates` and update the pom.xml file.

```
<properties>
  <solutionconstructs.version>2.5.0</solutionconstructs.version>
</properties>

<dependency>
    <groupId>software.amazon.awsconstructs</groupId>
    <artifactId>apigatewaylambda</artifactId>
    <version>${solutionconstructs.version}</version>
</dependency>
```

Run the command:

```
mvn install
```

Add an Amazon API Gateway/AWS Lambda pattern to your stack

Now, let's define the AWS Solutions Constructs pattern for implementing an Amazon API Gateway with an AWS Lambda proxy.

TypeScript

Replace the code in `lib/hello-constructs-stack.ts` with the following:

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import * as lambda from 'aws-cdk-lib/aws-lambda';
import * as api from 'aws-cdk-lib/aws-apigateway';
```

```
import { ApiGatewayToLambda, ApiGatewayToLambdaProps } from '@aws-solutions-
constructs/aws-apigateway-lambda';

export class HelloConstructsStack extends Stack {
  constructor(scope: Construct, id: string, props?: StackProps) {
    super(scope, id, props);

    // The code that defines your stack goes here
    const api_lambda_props: ApiGatewayToLambdaProps = {
      lambdaFunctionProps: {
        code: lambda.Code.fromAsset('lambda'),
        runtime: lambda.Runtime.NODEJS_14_X,
        handler: 'hello.handler'
      },
      apiGatewayProps: {
        defaultMethodOptions: {
          authorizationType: api.AuthorizationType.NONE
        }
      }
    };

    new ApiGatewayToLambda(this, 'ApiGatewayToLambda', api_lambda_props);
  }
}
```

Python

Replace the code in `hello_constructs/hello_constructs_stack.py` with the following:

```

from constructs import Construct
from aws_cdk import (
    aws_lambda as _lambda,
    aws_apigateway as apigw,
    App,
    Stack
)

from aws_solutions_constructs import (
    aws_apigateway_lambda as apigw_lambda
)

class HelloConstructsStack(Stack):
```

```
    def __init__(self, scope: Construct, id: str, **kwargs) -> None:
        super().__init__(scope, id, **kwargs)

        # The code that defines your stack goes here

        apigw_lambda.ApiGatewayToLambda(
            self, 'ApiGatewayToLambda',
            lambda_function_props=_lambda.FunctionProps(
                runtime=_lambda.Runtime.PYTHON_3_7,
                code=_lambda.Code.from_asset('lambda'),
                handler='hello.handler',
            ),
            api_gateway_props=apigw.RestApiProps(
                default_method_options=apigw.MethodOptions(
                    authorization_type=apigw.AuthorizationType.NONE
                )
            )
        )
```

Java

Replace the code in `HelloConstructsStack.java` with the following:

```
package com.myorg;

import software.constructs.Construct;
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;

import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awscdk.services.apigateway.*;
import software.amazon.awsconstructs.services.apigatewaylambda.ApiGatewayToLambda;
import
 software.amazon.awsconstructs.services.apigatewaylambda.ApiGatewayToLambdaProps;

public class HelloConstructsStack extends Stack {
    public HelloConstructsStack(final Construct scope, final String id) {
        this(scope, id, null);
    }

    public HelloConstructsStack(final Construct scope, final String id, final
 StackProps props) {
```

```
        super(scope, id, props);

        new ApiGatewayToLambda(this, "ApiGatewayToLambdaPattern", new
 ApiGatewayToLambdaProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_14_X) // execution environment
                        .code(Code.fromAsset("lambda")) // code loaded from the
 "lambda" directory
                        .handler("hello.handler") // file is "hello", function is
 "handler"
                        .build())
                .apiGatewayProps(new RestApiProps.Builder()
                        .defaultMethodOptions(new MethodOptions.Builder()
                                .authorizationType(AuthorizationType.NONE)
                                .build())
                        .build())
                .build());
    }
}
```

That’s it. This is all you need to do in order to define an API Gateway which proxies all requests to an AWS Lambda function. Let's compare our new stack to the original one:

TypeScript

```
npm run build
cdk diff
```

Python

```
cdk diff
```

Java

```
mvn package
cdk diff
```

The output should look like this:

```
Stack HelloConstructsStack
IAM Statement Changes
######################################################################################
#   # Resource       # Effect # Action          # Principal       # Condition        #
######################################################################################
# + # ${ApiGatewayTo # Allow  # lambda:InvokeF # Service:apigat # "ArnLike": {      #
#   # LambdaPattern/ #        # unction         # eway.amazonaws #   "AWS:SourceArn" #
#   # LambdaFunction #        #                 # .com            # : "arn:${AWS::Par #
#   # .Arn}          #        #                 #                 # tition}:execute-a #
#   #                #        #                 #                 # pi:${AWS::Region} #
#   #                #        #                 #                 # :${AWS::AccountId #
#   #                #        #                 #                 # }:${ApiGatewayToL #
#   #                #        #                 #                 # ambdaPatternLambd #
#   #                #        #                 #                 # aRestApiC0598E46} #
#   #                #        #                 #                 # /${ApiGatewayToLa #
#   #                #        #                 #                 # mbdaPattern/Lambd #
#   #                #        #                 #                 # aRestApi/Deployme #
#   #                #        #                 #                 # ntStage.prod}/*/* #
#   #                #        #                 #                 # "                 #
#   #                #        #                 #                 # }                 #
# + # ${ApiGatewayTo # Allow  # lambda:InvokeF # Service:apigat # "ArnLike": {      #
#   # LambdaPattern/ #        # unction         # eway.amazonaws #   "AWS:SourceArn" #
#   # LambdaFunction #        #                 # .com            # : "arn:${AWS::Par #
#   # .Arn}          #        #                 #                 # tition}:execute-a #
#   #                #        #                 #                 # pi:${AWS::Region} #
#   #                #        #                 #                 # :${AWS::AccountId #
#   #                #        #                 #                 # }:${ApiGatewayToL #
#   #                #        #                 #                 # ambdaPatternLambd #
#   #                #        #                 #                 # aRestApiC0598E46} #
#   #                #        #                 #                 # /test-invoke-stag #
#   #                #        #                 #                 # e/*/*"            #
#   #                #        #                 #                 # }                 #
# + # ${ApiGatewayTo # Allow  # lambda:InvokeF # Service:apigat # "ArnLike": {      #
#   # LambdaPattern/ #        # unction         # eway.amazonaws #   "AWS:SourceArn" #
#   # LambdaFunction #        #                 # .com            # : "arn:${AWS::Par #
#   # .Arn}          #        #                 #                 # tition}:execute-a #
#   #                #        #                 #                 # pi:${AWS::Region} #
#   #                #        #                 #                 # :${AWS::AccountId #
#   #                #        #                 #                 # }:${ApiGatewayToL #
#   #                #        #                 #                 # ambdaPatternLambd #
#   #                #        #                 #                 # aRestApiC0598E46} #
```

```
#   #                 #         #                 #                 # /${ApiGatewayToLa #
#   #                 #         #                 #                 # mbdaPattern/Lambd #
#   #                 #         #                 #                 # aRestApi/Deployme #
#   #                 #         #                 #                 # ntStage.prod}/*/" #
#   #                 #         #                 #                 # }                 #
# + # ${ApiGatewayTo # Allow   # lambda:InvokeF # Service:apigat # "ArnLike": {      #
#   # LambdaPattern/ #         # unction        # eway.amazonaws #   "AWS:SourceArn" #
#   # LambdaFunction #         #                # .com           # : "arn:${AWS::Par #
#   # .Arn}          #         #                #                # tition}:execute-a #
#   #                #         #                #                # pi:${AWS::Region} #
#   #                #         #                #                # :${AWS::AccountId #
#   #                #         #                #                # }:${ApiGatewayToL #
#   #                #         #                #                # ambdaPatternLambd #
#   #                #         #                #                # aRestApiC0598E46} #
#   #                #         #                #                # /test-invoke-stag #
#   #                #         #                #                # e/*/"             #
#   #                #         #                #                # }                 #
########################################################################################
# + # ${ApiGatewayTo # Allow   # sts:AssumeRole # Service:lambda #                   #
#   # LambdaPattern/ #         #                # .amazonaws.com #                   #
#   # LambdaFunction #         #                #                #                   #
#   # ServiceRole.Ar #         #                #                #                   #
#   # n}             #         #                #                #                   #
########################################################################################
# + # ${ApiGatewayTo # Allow   # sts:AssumeRole # Service:apigat #                   #
#   # LambdaPattern/ #         #                # eway.amazonaws #                   #
#   # LambdaRestApiC #         #                # .com           #                   #
#   # loudWatchRole. #         #                #                #                   #
#   # Arn}           #         #                #                #                   #
########################################################################################
# + # *              # Allow   # xray:PutTeleme # AWS:${ApiGatew #                   #
#   #                #         # tryRecords     # ayToLambdaPatt #                   #
#   #                #         # xray:PutTraceS # ern/LambdaFunc #                   #
#   #                #         # egments        # tionServiceRol #                   #
#   #                #         #                # e}             #                   #
########################################################################################
# + # arn:${AWS::Par # Allow   # logs:CreateLog # AWS:${ApiGatew #                   #
#   # tition}:logs:$ #         # Group          # ayToLambdaPatt #                   #
#   # {AWS::Region}: #         # logs:CreateLog # ern/LambdaRest #                   #
#   # ${AWS::Account #         # Stream         # ApiCloudWatchR #                   #
#   # Id}:*          #         # logs:DescribeL # ole}           #                   #
#   #                #         # ogGroups       #                #                   #
#   #                #         # logs:DescribeL #                #                   #
#   #                #         # ogStreams      #                #                   #
```

```
#   #                 #        # logs:FilterLog #                 #                    #
#   #                 #        # Events         #                 #                    #
#   #                 #        # logs:GetLogEve #                 #                    #
#   #                 #        # nts            #                 #                    #
#   #                 #        # logs:PutLogEve #                 #                    #
#   #                 #        # nts            #                 #                    #
#########################################################################################
# + # arn:${AWS::Par # Allow  # logs:CreateLog # AWS:${ApiGatew #                    #
#   # tition}:logs:$ #        # Group          # ayToLambdaPatt #                    #
#   # {AWS::Region}: #        # logs:CreateLog # ern/LambdaFunc #                    #
#   # ${AWS::Account #        # Stream         # tionServiceRol #                    #
#   # Id}:log-group: #        # logs:PutLogEve # e}             #                    #
#   # /aws/lambda/*  #        # nts            #                #                    #
#########################################################################################
(NOTE: There may be security-related changes not in this list. See https://github.com/
aws/aws-cdk/issues/1299)

Parameters
[+] Parameter BootstrapVersion BootstrapVersion:
 {"Type":"AWS::SSM::Parameter::Value<String>","Default":"/cdk-bootstrap/hnb659fds/
version","Description":"Version of the CDK Bootstrap resources in this environment,
 automatically retrieved from SSM Parameter Store. [cdk:skip]"}

Conditions
[+] Condition CDKMetadata/Condition CDKMetadataAvailable: {"Fn::Or":
[{"Fn::Or":[{"Fn::Equals":[{"Ref":"AWS::Region"},"af-south-1"]},{"Fn::Equals":
[{"Ref":"AWS::Region"},"ap-east-1"]},{"Fn::Equals":[{"Ref":"AWS::Region"},"ap-
northeast-1"]},{"Fn::Equals":[{"Ref":"AWS::Region"},"ap-northeast-2"]},{"Fn::Equals":
[{"Ref":"AWS::Region"},"ap-south-1"]},{"Fn::Equals":[{"Ref":"AWS::Region"},"ap-
southeast-1"]},{"Fn::Equals":[{"Ref":"AWS::Region"},"ap-southeast-2"]},
{"Fn::Equals":[{"Ref":"AWS::Region"},"ca-central-1"]},{"Fn::Equals":
[{"Ref":"AWS::Region"},"cn-north-1"]},{"Fn::Equals":[{"Ref":"AWS::Region"},"cn-
northwest-1"]}]},{"Fn::Or":[{"Fn::Equals":[{"Ref":"AWS::Region"},"eu-
central-1"]},{"Fn::Equals":[{"Ref":"AWS::Region"},"eu-north-1"]},{"Fn::Equals":
[{"Ref":"AWS::Region"},"eu-south-1"]},{"Fn::Equals":[{"Ref":"AWS::Region"},"eu-
west-1"]},{"Fn::Equals":[{"Ref":"AWS::Region"},"eu-west-2"]},{"Fn::Equals":
[{"Ref":"AWS::Region"},"eu-west-3"]},{"Fn::Equals":[{"Ref":"AWS::Region"},"me-
south-1"]},{"Fn::Equals":[{"Ref":"AWS::Region"},"sa-east-1"]},{"Fn::Equals":
[{"Ref":"AWS::Region"},"us-east-1"]},{"Fn::Equals":[{"Ref":"AWS::Region"},"us-
east-2"]}]},{"Fn::Or":[{"Fn::Equals":[{"Ref":"AWS::Region"},"us-west-1"]},
{"Fn::Equals":[{"Ref":"AWS::Region"},"us-west-2"]}]}]}

Resources
```

```
[+] AWS::IAM::Role ApiGatewayToLambdaPattern/LambdaFunctionServiceRole
 ApiGatewayToLambdaPatternLambdaFunctionServiceRole0C123D8D
[+] AWS::IAM::Policy ApiGatewayToLambdaPattern/LambdaFunctionServiceRole/DefaultPolicy
 ApiGatewayToLambdaPatternLambdaFunctionServiceRoleDefaultPolicy253751F2
[+] AWS::Lambda::Function ApiGatewayToLambdaPattern//LambdaFunction
 ApiGatewayToLambdaPatternLambdaFunction5DC51B7E
[+] AWS::Logs::LogGroup ApiGatewayToLambdaPattern/ApiAccessLogGroup
 ApiGatewayToLambdaPatternApiAccessLogGroup9383E9FC
[+] AWS::ApiGateway::RestApi ApiGatewayToLambdaPattern/LambdaRestApi
 ApiGatewayToLambdaPatternLambdaRestApiC0598E46
[+] AWS::ApiGateway::Deployment ApiGatewayToLambdaPattern/LambdaRestApi/Deployment
 ApiGatewayToLambdaPatternLambdaRestApiDeployment4109DB93346ab5d96a64d161f4cf4f020d3cdf94
[+] AWS::ApiGateway::Stage ApiGatewayToLambdaPattern/LambdaRestApi/DeploymentStage.prod
 ApiGatewayToLambdaPatternLambdaRestApiDeploymentStageprodFDEB8074
[+] AWS::ApiGateway::Resource ApiGatewayToLambdaPattern/LambdaRestApi/Default/{proxy+}
 ApiGatewayToLambdaPatternLambdaRestApiproxyF368A2D5
[+] AWS::Lambda::Permission ApiGatewayToLambdaPattern/LambdaRestApi/Default/{proxy+}/
ANY/
ApiPermission.HelloConstructsStackApiGatewayToLambdaPatternLambdaRestApi553584F5.ANY..
{proxy+}
 ApiGatewayToLambdaPatternLambdaRestApiproxyANYApiPermissionHelloConstructsStackApiGatewayToLam
[+] AWS::Lambda::Permission ApiGatewayToLambdaPattern/LambdaRestApi/Default/{proxy+}/
ANY/
ApiPermission.Test.HelloConstructsStackApiGatewayToLambdaPatternLambdaRestApi553584F5.ANY..
{proxy+}
 ApiGatewayToLambdaPatternLambdaRestApiproxyANYApiPermissionTestHelloConstructsStackApiGatewayT
[+] AWS::ApiGateway::Method ApiGatewayToLambdaPattern/LambdaRestApi/Default/{proxy+}/
ANY ApiGatewayToLambdaPatternLambdaRestApiproxyANY321FD2C2
[+] AWS::Lambda::Permission ApiGatewayToLambdaPattern/LambdaRestApi/Default/ANY/
ApiPermission.HelloConstructsStackApiGatewayToLambdaPatternLambdaRestApi553584F5.ANY..
 ApiGatewayToLambdaPatternLambdaRestApiANYApiPermissionHelloConstructsStackApiGatewayToLambdaPa
[+] AWS::Lambda::Permission ApiGatewayToLambdaPattern/LambdaRestApi/Default/ANY/
ApiPermission.Test.HelloConstructsStackApiGatewayToLambdaPatternLambdaRestApi553584F5.ANY..
 ApiGatewayToLambdaPatternLambdaRestApiANYApiPermissionTestHelloConstructsStackApiGatewayToLamb
[+] AWS::ApiGateway::Method ApiGatewayToLambdaPattern/LambdaRestApi/Default/ANY
 ApiGatewayToLambdaPatternLambdaRestApiANY07ADEFED
[+] AWS::ApiGateway::UsagePlan ApiGatewayToLambdaPattern/LambdaRestApi/UsagePlan
 ApiGatewayToLambdaPatternLambdaRestApiUsagePlan837CF24C
[+] AWS::IAM::Role ApiGatewayToLambdaPattern/LambdaRestApiCloudWatchRole
 ApiGatewayToLambdaPatternLambdaRestApiCloudWatchRole3977FB08
[+] AWS::ApiGateway::Account ApiGatewayToLambdaPattern/LambdaRestApiAccount
 ApiGatewayToLambdaPatternLambdaRestApiAccountA31D1168

Outputs
```

```
[+] Output ApiGatewayToLambdaPattern/LambdaRestApi/Endpoint
 ApiGatewayToLambdaPatternLambdaRestApiEndpoint9A7AA3AD: {"Value":{"Fn::Join":["",
["https://",{"Ref":"ApiGatewayToLambdaPatternLambdaRestApiC0598E46"},".execute-
api.",{"Ref":"AWS::Region"},".",{"Ref":"AWS::URLSuffix"},"/",
{"Ref":"ApiGatewayToLambdaPatternLambdaRestApiDeploymentStageprodFDEB8074"},"/"]]}}

Other Changes
[+] Unknown Rules: {"CheckBootstrapVersion":{"Assertions":
[{"Assert":{"Fn::Not":[{"Fn::Contains":[["1","2","3","4","5"],
{"Ref":"BootstrapVersion"}]}]},"AssertDescription":"CDK bootstrap stack version 6
 required. Please run 'cdk bootstrap' with a recent version of the CDK CLI."}]}}
```

This simple example with one well-architected pattern from the AWS Solutions Constructs added 17 new resources to your stack.

cdk deploy

Tip

Before you can deploy your first AWS CDK app containing a Lambda function, you must bootstrap your AWS environment. This creates a staging bucket that the AWS CDK uses to deploy stacks containing assets. If this is the first time you are using the AWS CDK to deploy assets, you will need to run the `cdk bootstrap` command to deploy the CDK toolkit stack into your AWS environment.

Okay, ready to deploy?

```
cdk deploy
```

Stack outputs

When deployment is complete, you'll notice this line:

```
Outputs:
  HelloConstructsStack.RestApiEndpoint0551178A = https://
```

```
    xxxxxxxxxx
    .execute-api.us-east-1.amazonaws.com/prod/
```

This is a stack output that's automatically added by the AWS Solutions Constructs pattern and includes the URL of the API Gateway endpoint.

Testing your app

Let's try to hit this endpoint with `curl`. Copy the URL and execute (your prefix and Region will likely be different).

```
curl https://
   xxxxxxxxxx
    .execute-api.us-east-1.amazonaws.com/prod/
```

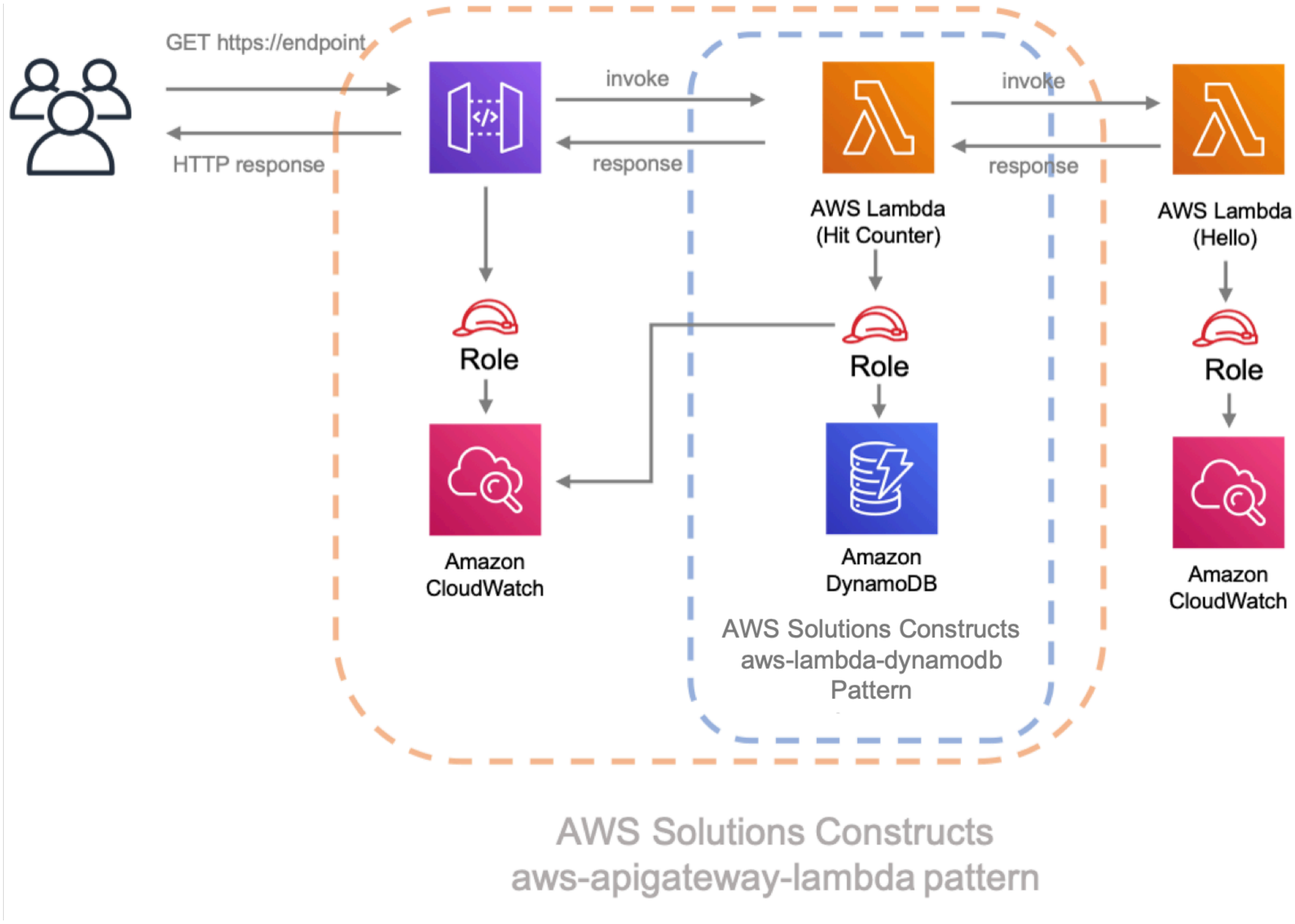
Output should look like this:

```
Hello, AWS Solutions Constructs! You've hit /
```

If this is the output you received, your app works!

Walkthrough - Part 2

This tutorial walks you through how to modify the "Hello Constructs" app created in part 1 . Our modification will add a site hit counter using the AWS Lambda to DynamoDB pattern from AWS Solutions Constructs. Modifying the Hello Constructs app will result in the following solution:



Hit Counter Lambda code

Let's get started by writing the code for the Hit Counter AWS Lambda function. This function will:

- increment a counter related to the API path in a Amazon DynamoDB table,
- invoke the downstream Hello AWS Lambda function,
- and return the response to end user.

TypeScript

Add a file called `lambda/hitcounter.js` with the following contents:

```
const { DynamoDB, Lambda } = require('aws-sdk');
```

```
exports.handler = async function(event) {
  console.log("request:", JSON.stringify(event, undefined, 2));

  // create AWS SDK clients
  const dynamo = new DynamoDB();
  const lambda = new Lambda();

  // update dynamo entry for "path" with hits++
  await dynamo.updateItem({
    TableName: process.env.DDB_TABLE_NAME,
    Key: { path: { S: event.path } },
    UpdateExpression: 'ADD hits :incr',
    ExpressionAttributeValues: { ':incr': { N: '1' } }
  }).promise();

  // call downstream function and capture response
  const resp = await lambda.invoke({
    FunctionName: process.env.DOWNSTREAM_FUNCTION_NAME,
    Payload: JSON.stringify(event)
  }).promise();

  console.log('downstream response:', JSON.stringify(resp, undefined, 2));

  // return response back to upstream caller
  return JSON.parse(resp.Payload);
};
```

Python

Add a file called `lambda/hitcounter.py` with the following contents:

```
import json
import os
import boto3

ddb = boto3.resource('dynamodb')
table = ddb.Table(os.environ['DDB_TABLE_NAME'])
_lambda = boto3.client('lambda')


def handler(event, context):
    print('request: {}'.format(json.dumps(event)))
    table.update_item(
```

```
        Key={'path': event['path']},
        UpdateExpression='ADD hits :incr',
        ExpressionAttributeValues={':incr': 1}
    )

    resp = _lambda.invoke(
        FunctionName=os.environ['DOWNSTREAM_FUNCTION_NAME'],
        Payload=json.dumps(event),
    )

    body = resp['Payload'].read()

    print('downstream response: {}'.format(body))
    return json.loads(body)
```

Java

Add a file called `lambda/hitcounter.js` with the following contents:

```
const { DynamoDB, Lambda } = require('aws-sdk');

exports.handler = async function(event) {
  console.log("request:", JSON.stringify(event, undefined, 2));

  // create AWS SDK clients
  const dynamo = new DynamoDB();
  const lambda = new Lambda();

  // update dynamo entry for "path" with hits++
  await dynamo.updateItem({
    TableName: process.env.DDB_TABLE_NAME,
    Key: { path: { S: event.path } },
    UpdateExpression: 'ADD hits :incr',
    ExpressionAttributeValues: { ':incr': { N: '1' } }
  }).promise();

  // call downstream function and capture response
  const resp = await lambda.invoke({
    FunctionName: process.env.DOWNSTREAM_FUNCTION_NAME,
    Payload: JSON.stringify(event)
  }).promise();

  console.log('downstream response:', JSON.stringify(resp, undefined, 2));
```

```
  // return response back to upstream caller
  return JSON.parse(resp.Payload);
};
```

Install the new dependency

As usual, we first need to install the dependency we need for our solution update. Install the AWS Solutions Constructs `aws-lambda-dynamodb` module and all its dependency into our project:

TypeScript

```
npm install -s @aws-solutions-constructs/aws-lambda-dynamodb
```

Python

```
pip install aws_solutions_constructs.aws_lambda_dynamodb
```

Java

Edit the `pom.xml` file with the following information:

```
<dependency>
    <groupId>software.amazon.awsconstructs</groupId>
    <artifactId>lambdadynamodb</artifactId>
    <version>${solutionconstructs.version}</version>
</dependency>
```

Run the command:

```
mvn install
```

Define the resources

Now, let's update our stack code to accomodate our new architecture.

First, we are going to import our new dependency and move the "Hello" function outside of the `aws-apigateway-lambda` pattern we created in part 1.

TypeScript

Replace the code in `lib/hello-constructs-stack.ts` with the following:

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import * as lambda from 'aws-cdk-lib/aws-lambda';
import * as api from 'aws-cdk-lib/aws-apigateway';
import * as dynamodb from 'aws-cdk-lib/aws-dynamodb';
import { ApiGatewayToLambda, ApiGatewayToLambdaProps } from '@aws-solutions-
constructs/aws-apigateway-lambda';
import { LambdaToDynamoDB, LambdaToDynamoDBProps } from '@aws-solutions-constructs/
aws-lambda-dynamodb';

export class HelloConstructsStack extends Stack {
  constructor(scope: Construct, id: string, props?: StackProps) {
    super(scope, id, props);

    // The code that defines your stack goes here

    const helloFunc = new lambda.Function(this, 'HelloHandler', {
      runtime: lambda.Runtime.NODEJS_12_X,
      code: lambda.Code.fromAsset('lambda'),
      handler: 'hello.handler'
    });

    const api_lambda_props: ApiGatewayToLambdaProps = {
      lambdaFunctionProps: {
        code: lambda.Code.fromAsset('lambda'),
        runtime: lambda.Runtime.NODEJS_12_X,
        handler: 'hello.handler'
      },
      apiGatewayProps: {
        defaultMethodOptions: {
          authorizationType: api.AuthorizationType.NONE
        }
```

```
      }
    };

    new ApiGatewayToLambda(this, 'ApiGatewayToLambda', api_lambda_props);
  }
}
```

Python

Replace the code in `hello_constructs/hello_constructs_stack.py` with the following:

```
from constructs import Construct
from aws_cdk import (
    aws_lambda as _lambda,
    aws_apigateway as apigw,
    aws_dynamodb as ddb,
    App,
    Stack
)
from aws_solutions_constructs import (
    aws_apigateway_lambda as apigw_lambda,
    aws_lambda_dynamodb as lambda_ddb
)

class HelloConstructsStack(Stack):

    def __init__(self, scope: Construct, id: str, **kwargs) -> None:
        super().__init__(scope, id, **kwargs)

        # The code that defines your stack goes here

        self._handler = _lambda.Function(
          self, 'HelloHandler',
          runtime=_lambda.Runtime.PYTHON_3_7,
          handler='hello.handler',
          code=_lambda.Code.from_asset('lambda'),
        )

        apigw_lambda.ApiGatewayToLambda(
            self, 'ApiGatewayToLambda',
            lambda_function_props=_lambda.FunctionProps(
                runtime=_lambda.Runtime.PYTHON_3_7,
                code=_lambda.Code.from_asset('lambda'),
```

```
                handler='hello.handler',
            ),
            api_gateway_props=apigw.RestApiProps(
                default_method_options=apigw.MethodOptions(
                    authorization_type=apigw.AuthorizationType.NONE
                )
            )
        )
```

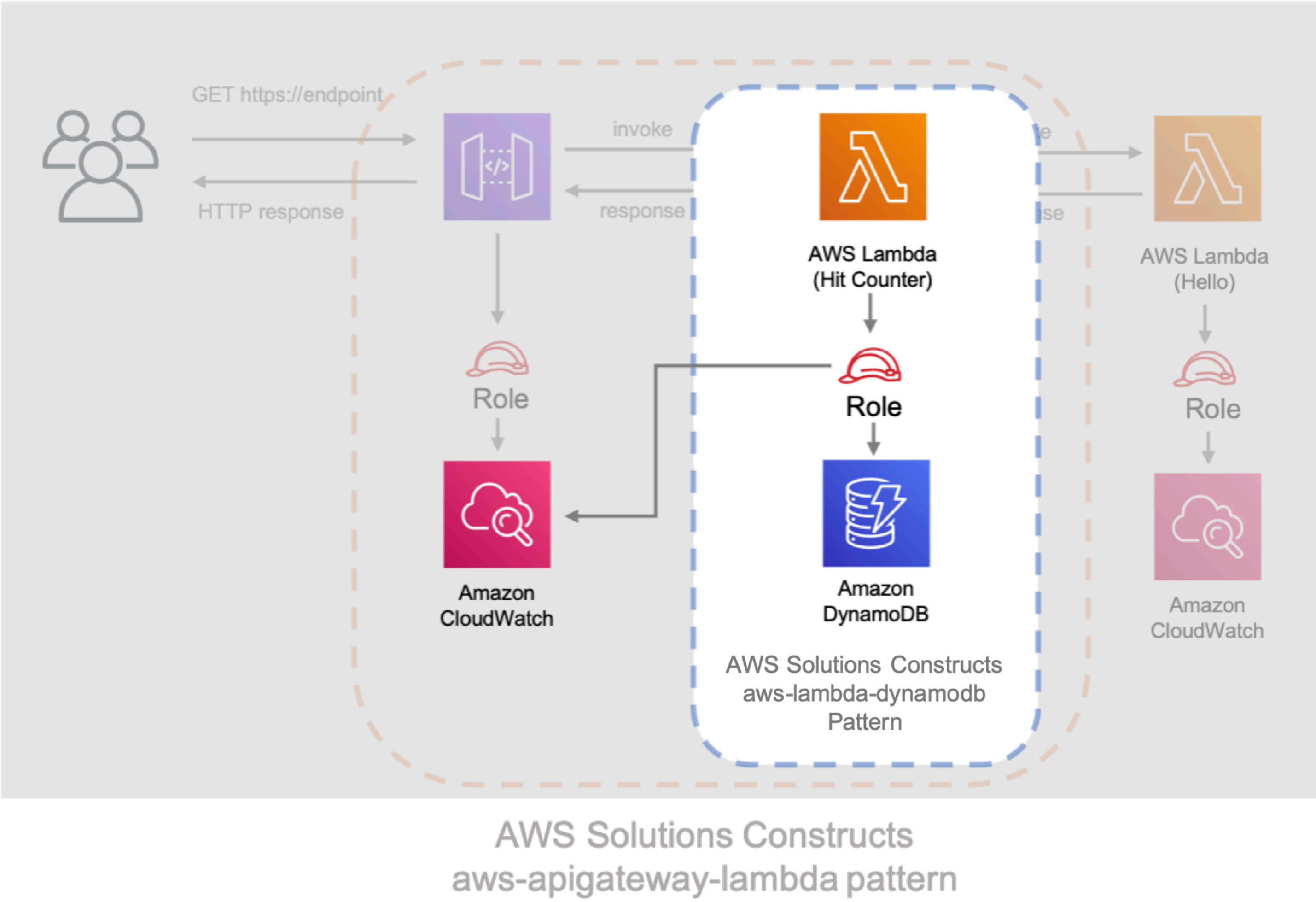
Java

Replace the code in `HelloConstructsStack.java` with the following:

```
package com.myorg;

import software.constructs.Construct;
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;

import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awscdk.services.apigateway.*;
import software.amazon.awsconstructs.services.apigatewaylambda.ApiGatewayToLambda;
import
 software.amazon.awsconstructs.services.apigatewaylambda.ApiGatewayToLambdaProps;

public class HelloConstructsStack extends Stack {
    public HelloConstructsStack(final Construct scope, final String id) {
        this(scope, id, null);
    }

    public HelloConstructsStack(final Construct scope, final String id, final
 StackProps props) {
        super(scope, id, props);

        final Function hello = Function.Builder.create(this, "HelloHandler")
                .runtime(Runtime.NODEJS_14_X) // execution environment
                .code(Code.fromAsset("lambda")) // code loaded from the "lambda"
 directory
                .handler("hello.handler") // file is "hello", function is "handler"
                .build();
```

```
        new ApiGatewayToLambda(this, "ApiGatewayToLambdaPattern", new
 ApiGatewayToLambdaProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_14_X) // execution environment
                        .code(Code.fromAsset("lambda")) // code loaded from the
 "lambda" directory
                        .handler("hello.handler") // file is "hello", function is
 "handler"
                        .build())
                .apiGatewayProps(new RestApiProps.Builder()
                        .defaultMethodOptions(new MethodOptions.Builder()
                                .authorizationType(AuthorizationType.NONE)
                                .build())
                        .build())
                .build());
    }
}
```

Next, we are going to add the `aws-lambda-dynamodb` pattern to build out the hit counter service for our updated architecture.



The next update below defines the properties for the `aws-lambda-dynamodb` pattern by defining the AWS Lambda function with the Hit Counter handler. Additionally, the Amazon DynamoDB table is defined with a name of `SolutionsConstructsHits` and a partition key of `path` .

TypeScript

Edit the file `lib/hello-constructs-stack.ts` with the following:

```
import { Construct } from 'constructs';
import { Stack, StackProps, RemovalPolicy } from 'aws-cdk-lib';
import * as lambda from 'aws-cdk-lib/aws-lambda';
import * as api from 'aws-cdk-lib/aws-apigateway';
import * as dynamodb from 'aws-cdk-lib/aws-dynamodb';
import { ApiGatewayToLambda, ApiGatewayToLambdaProps } from '@aws-solutions-constructs/aws-apigateway-lambda';
```

```
import { LambdaToDynamoDB, LambdaToDynamoDBProps } from '@aws-solutions-constructs/
aws-lambda-dynamodb';

export class HelloConstructsStack extends Stack {
  constructor(scope: Construct, id: string, props?: StackProps) {
    super(scope, id, props);

    // The code that defines your stack goes here

    const helloFunc = new lambda.Function(this, 'HelloHandler', {
      runtime: lambda.Runtime.NODEJS_12_X,
      code: lambda.Code.fromAsset('lambda'),
      handler: 'hello.handler'
    });

    // hit counter, aws-lambda-dynamodb pattern
    const lambda_ddb_props: LambdaToDynamoDBProps = {
      lambdaFunctionProps: {
          code: lambda.Code.fromAsset(`lambda`),
          runtime: lambda.Runtime.NODEJS_12_X,
          handler: 'hitcounter.handler',
          environment: {
              DOWNSTREAM_FUNCTION_NAME: helloFunc.functionName
          }
      },
      dynamoTableProps: {
          tableName: 'SolutionsConstructsHits',
          partitionKey: { name: 'path', type: dynamodb.AttributeType.STRING },
          removalPolicy: RemovalPolicy.DESTROY
      }
    };

    const hitcounter = new LambdaToDynamoDB(this, 'LambdaToDynamoDB',
 lambda_ddb_props);

    const api_lambda_props: ApiGatewayToLambdaProps = {
      lambdaFunctionProps: {
        code: lambda.Code.fromAsset('lambda'),
        runtime: lambda.Runtime.NODEJS_12_X,
        handler: 'hello.handler'
      },
      apiGatewayProps: {
        defaultMethodOptions: {
          authorizationType: api.AuthorizationType.NONE
```

```
        }
      }
    };

    new ApiGatewayToLambda(this, 'ApiGatewayToLambda', api_lambda_props);
  }
}
```

Python

Edit the file `hello_constructs/hello_constructs_stack.py` with the following:

```
from constructs import Construct
from aws_cdk import (
    aws_lambda as _lambda,
    aws_apigateway as apigw,
    aws_dynamodb as ddb,
    App,
    Stack,
    RemovalPolicy
)
from aws_solutions_constructs import (
    aws_apigateway_lambda as apigw_lambda,
    aws_lambda_dynamodb as lambda_ddb
)


class HelloConstructsStack(Stack):

    def __init__(self, scope: Construct, id: str, **kwargs) -> None:
        super().__init__(scope, id, **kwargs)

        # The code that defines your stack goes here

        self.hello_func = _lambda.Function(
            self, 'HelloHandler',
            runtime=_lambda.Runtime.PYTHON_3_7,
            handler='hello.handler',
            code=_lambda.Code.from_asset('lambda'),
        )

        #  hit counter, aws-lambda-dynamodb pattern
        self.hit_counter = lambda_ddb.LambdaToDynamoDB(
```

```
            self, 'LambdaToDynamoDB',
            lambda_function_props=_lambda.FunctionProps(
                runtime=_lambda.Runtime.PYTHON_3_7,
                code=_lambda.Code.from_asset('lambda'),
                handler='hitcounter.handler',
                environment={
                    'DOWNSTREAM_FUNCTION_NAME': self.hello_func.function_name
                }
            ),
            dynamo_table_props=ddb.TableProps(
                table_name='SolutionsConstructsHits',
                partition_key={
                    'name': 'path',
                    'type': ddb.AttributeType.STRING
                },
                removal_policy=RemovalPolicy.DESTROY
            )
        )

        apigw_lambda.ApiGatewayToLambda(
            self, 'ApiGatewayToLambda',
            lambda_function_props=_lambda.FunctionProps(
                runtime=_lambda.Runtime.PYTHON_3_7,
                code=_lambda.Code.from_asset('lambda'),
                handler='hello.handler',
            ),
            api_gateway_props=apigw.RestApiProps(
                default_method_options=apigw.MethodOptions(
                    authorization_type=apigw.AuthorizationType.NONE
                )
            )
        )
```

Java

Edit the file `HelloConstructsStack.java` with the following:

```
package com.myorg;

import java.util.Map;
import java.util.HashMap;
```

```
import software.constructs.Construct;
import software.amazon.awscdk.RemovalPolicy;
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;

import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awscdk.services.apigateway.*;
import software.amazon.awscdk.services.dynamodb.*;
import software.amazon.awsconstructs.services.apigatewaylambda.ApiGatewayToLambda;
import
 software.amazon.awsconstructs.services.apigatewaylambda.ApiGatewayToLambdaProps;
import software.amazon.awsconstructs.services.lambdadynamodb.*;

public class HelloConstructsStack extends Stack {
    public HelloConstructsStack(final Construct scope, final String id) {
        this(scope, id, null);
    }

    public HelloConstructsStack(final Construct scope, final String id, final
 StackProps props) {
        super(scope, id, props);

        final Function helloFunc = Function.Builder.create(this, "HelloHandler")
                .runtime(Runtime.NODEJS_14_X) // execution environment
                .code(Code.fromAsset("lambda")) // code loaded from the "lambda"
 directory
                .handler("hello.handler") // file is "hello", function is "handler"
                .build();

        final Map<String, String> lambdaEnvironment = new HashMap<>();
        lambdaEnvironment.put("DOWNSTREAM_FUNCTION_NAME",
 helloFunc.getFunctionName());

        final LambdaToDynamoDB hitcounter = new LambdaToDynamoDB(this,
 "LambdaToDynamoDBPattern",
                new LambdaToDynamoDBProps.Builder()
                        .lambdaFunctionProps(new FunctionProps.Builder()
                                .runtime(Runtime.NODEJS_14_X) // execution
 environment
                                .code(Code.fromAsset("lambda")) // code loaded from
 the "lambda" directory
                                .handler("hitcounter.handler") // file is "hello",
 function is "handler"
```

```
                                .environment(lambdaEnvironment)
                                .build())
                        .dynamoTableProps(new TableProps.Builder()
                                .tableName("SolutionsConstructsHits")
                                .partitionKey(new Attribute.Builder()
                                        .name("path")
                                        .type(AttributeType.STRING)
                                        .build())
                                .removalPolicy(RemovalPolicy.DESTROY)
                                .build())
                        .build());

        new ApiGatewayToLambda(this, "ApiGatewayToLambdaPattern", new
 ApiGatewayToLambdaProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_14_X) // execution environment
                        .code(Code.fromAsset("lambda")) // code loaded from the
 "lambda" directory
                        .handler("hello.handler") // file is "hello", function is
 "handler"
                        .build())
                .apiGatewayProps(new RestApiProps.Builder()
                        .defaultMethodOptions(new MethodOptions.Builder()
                                .authorizationType(AuthorizationType.NONE)
                                .build())
                        .build())
                .build());
    }
}
```

Next, we need to grant the Hit Counter function created from the `aws-lambda-dynamodb` pattern added above permission to invoke our Hello function.

TypeScript

Edit the file `lib/hello-constructs-stack.ts` with the following:

```
import { Construct } from 'constructs';
import { Stack, StackProps, RemovalPolicy } from 'aws-cdk-lib';
import * as lambda from 'aws-cdk-lib/aws-lambda';
```

```
import * as api from 'aws-cdk-lib/aws-apigateway';
import * as dynamodb from 'aws-cdk-lib/aws-dynamodb';
import { ApiGatewayToLambda, ApiGatewayToLambdaProps } from '@aws-solutions-
constructs/aws-apigateway-lambda';
import { LambdaToDynamoDB, LambdaToDynamoDBProps } from '@aws-solutions-constructs/
aws-lambda-dynamodb';

  export class HelloConstructsStack extends Stack {
    constructor(scope: Construct, id: string, props?: StackProps) {
      super(scope, id, props);

      // The code that defines your stack goes here

      // hello function responding to http requests
      const helloFunc = new lambda.Function(this, 'HelloHandler', {
        runtime: lambda.Runtime.NODEJS_12_X,
        code: lambda.Code.fromAsset('lambda'),
        handler: 'hello.handler'
      });

      // hit counter, aws-lambda-dynamodb pattern
      const lambda_ddb_props: LambdaToDynamoDBProps = {
        lambdaFunctionProps: {
            code: lambda.Code.fromAsset(`lambda`),
            runtime: lambda.Runtime.NODEJS_12_X,
            handler: 'hitcounter.handler',
            environment: {
                DOWNSTREAM_FUNCTION_NAME: helloFunc.functionName
            }
        },
        dynamoTableProps: {
            tableName: 'SolutionsConstructsHits',
            partitionKey: { name: 'path', type: dynamodb.AttributeType.STRING },
            removalPolicy: RemovalPolicy.DESTROY
        }
      };

      const hitcounter = new LambdaToDynamoDB(this, 'LambdaToDynamoDB',
 lambda_ddb_props);

      // grant the hitcounter lambda role invoke permissions to the hello function
      helloFunc.grantInvoke(hitcounter.lambdaFunction);

      const api_lambda_props: ApiGatewayToLambdaProps = {
```

```
        lambdaFunctionProps: {
          code: lambda.Code.fromAsset('lambda'),
          runtime: lambda.Runtime.NODEJS_12_X,
          handler: 'hello.handler'
        },
        apiGatewayProps: {
          defaultMethodOptions: {
            authorizationType: api.AuthorizationType.NONE
          }
        }
      };

      new ApiGatewayToLambda(this, 'ApiGatewayToLambda', api_lambda_props);
    }
  }
```

Python

Edit the file `hello_constructs/hello_constructs_stack.py` with the following:

```
from constructs import Construct
from aws_cdk import (
    aws_lambda as _lambda,
    aws_apigateway as apigw,
    aws_dynamodb as ddb,
    App,
    Stack,
    RemovalPolicy
)

from aws_solutions_constructs import (
    aws_apigateway_lambda as apigw_lambda,
    aws_lambda_dynamodb as lambda_ddb
)

class HelloConstructsStack(Stack):

    def __init__(self, scope: Construct, id: str, **kwargs) -> None:
        super().__init__(scope, id, **kwargs)

        # The code that defines your stack goes here

        self.hello_func = _lambda.Function(
```

```
            self, 'HelloHandler',
            runtime=_lambda.Runtime.PYTHON_3_7,
            handler='hello.handler',
            code=_lambda.Code.from_asset('lambda'),
          )

          #  hit counter, aws-lambda-dynamodb pattern
          self.hit_counter = lambda_ddb.LambdaToDynamoDB(
              self, 'LambdaToDynamoDB',
              lambda_function_props=_lambda.FunctionProps(
                  runtime=_lambda.Runtime.PYTHON_3_7,
                  code=_lambda.Code.from_asset('lambda'),
                  handler='hitcounter.handler',
                  environment={
                      'DOWNSTREAM_FUNCTION_NAME': self.hello_func.function_name
                  }
              ),
              dynamo_table_props=ddb.TableProps(
                  table_name='SolutionsConstructsHits',
                  partition_key={
                      'name': 'path',
                      'type': ddb.AttributeType.STRING
                  },
                  removal_policy=RemovalPolicy.DESTROY
              )
          )

          # grant the hitcounter lambda role invoke permissions to the hello function
          self.hello_func.grant_invoke(self.hit_counter.lambda_function)

          apigw_lambda.ApiGatewayToLambda(
              self, 'ApiGatewayToLambda',
              lambda_function_props=_lambda.FunctionProps(
                  runtime=_lambda.Runtime.PYTHON_3_7,
                  code=_lambda.Code.from_asset('lambda'),
                  handler='hello.handler',
              ),
              api_gateway_props=apigw.RestApiProps(
                  default_method_options=apigw.MethodOptions(
                      authorization_type=apigw.AuthorizationType.NONE
                  )
              )
          )
```

Java

Edit the file `src/../HelloConstructsStack.java` with the following:

```
package com.myorg;

import java.util.Map;
import java.util.HashMap;

import software.constructs.Construct;
import software.amazon.awscdk.RemovalPolicy;
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;

import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awscdk.services.apigateway.*;
import software.amazon.awscdk.services.dynamodb.*;
import software.amazon.awsconstructs.services.apigatewaylambda.ApiGatewayToLambda;
import
 software.amazon.awsconstructs.services.apigatewaylambda.ApiGatewayToLambdaProps;
import software.amazon.awsconstructs.services.lambdadynamodb.*;

public class HelloConstructsStack extends Stack {
    public HelloConstructsStack(final Construct scope, final String id) {
        this(scope, id, null);
    }

    public HelloConstructsStack(final Construct scope, final String id, final
 StackProps props) {
        super(scope, id, props);

        final Function helloFunc = Function.Builder.create(this, "HelloHandler")
                .runtime(Runtime.NODEJS_14_X) // execution environment
                .code(Code.fromAsset("lambda")) // code loaded from the "lambda"
 directory
                .handler("hello.handler") // file is "hello", function is "handler"
                .build();

        final Map<String, String> lambdaEnvironment = new HashMap<>();
        lambdaEnvironment.put("DOWNSTREAM_FUNCTION_NAME",
 helloFunc.getFunctionName());
```

```
        final LambdaToDynamoDB hitcounter = new LambdaToDynamoDB(this,
 "LambdaToDynamoDBPattern",
                new LambdaToDynamoDBProps.Builder()
                        .lambdaFunctionProps(new FunctionProps.Builder()
                                .runtime(Runtime.NODEJS_14_X) // execution
 environment
                                .code(Code.fromAsset("lambda")) // code loaded from
 the "lambda" directory
                                .handler("hitcounter.handler") // file is "hello",
 function is "handler"
                                .environment(lambdaEnvironment)
                                .build())
                        .dynamoTableProps(new TableProps.Builder()
                                .tableName("SolutionsConstructsHits")
                                .partitionKey(new Attribute.Builder()
                                        .name("path")
                                        .type(AttributeType.STRING)
                                        .build())
                                .removalPolicy(RemovalPolicy.DESTROY)
                                .build())
                        .build());

        // grant the hitcounter lambda role invoke permissions to the hello function
        helloFunc.grantInvoke(hitcounter.getLambdaFunction());

        new ApiGatewayToLambda(this, "ApiGatewayToLambdaPattern", new
 ApiGatewayToLambdaProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_14_X) // execution environment
                        .code(Code.fromAsset("lambda")) // code loaded from the
 "lambda" directory
                        .handler("hello.handler") // file is "hello", function is
 "handler"
                        .build())
                .apiGatewayProps(new RestApiProps.Builder()
                        .defaultMethodOptions(new MethodOptions.Builder()
                                .authorizationType(AuthorizationType.NONE)
                                .build())
                        .build())
                .build());
    }
}
```

Finally, we need to update our original `aws-apigateway-lambda` pattern to utilize our new Hit Counter function that was provisioned with the `aws-lambda-dynamodb` pattern above.

TypeScript

Edit the file `lib/hello-constructs-stack.ts` with the following:

```
import { Construct } from 'constructs';
import { Stack, StackProps, RemovalPolicy } from 'aws-cdk-lib';
import * as lambda from 'aws-cdk-lib/aws-lambda';
import * as api from 'aws-cdk-lib/aws-apigateway';
import * as dynamodb from 'aws-cdk-lib/aws-dynamodb';
import { ApiGatewayToLambda, ApiGatewayToLambdaProps } from '@aws-solutions-
constructs/aws-apigateway-lambda';
import { LambdaToDynamoDB, LambdaToDynamoDBProps } from '@aws-solutions-constructs/
aws-lambda-dynamodb';

export class HelloConstructsStack extends Stack {
  constructor(scope: Construct, id: string, props?: StackProps) {
    super(scope, id, props);

    // The code that defines your stack goes here

    // hello function responding to http requests
    const helloFunc = new lambda.Function(this, 'HelloHandler', {
      runtime: lambda.Runtime.NODEJS_12_X,
      code: lambda.Code.fromAsset('lambda'),
      handler: 'hello.handler'
    });

    // hit counter, aws-lambda-dynamodb pattern
    const lambda_ddb_props: LambdaToDynamoDBProps = {
      lambdaFunctionProps: {
        code: lambda.Code.fromAsset(`lambda`),
        runtime: lambda.Runtime.NODEJS_12_X,
        handler: 'hitcounter.handler',
        environment: {
          DOWNSTREAM_FUNCTION_NAME: helloFunc.functionName
        }
```

```
      },
      dynamoTableProps: {
        tableName: 'SolutionsConstructsHits',
        partitionKey: { name: 'path', type: dynamodb.AttributeType.STRING },
        removalPolicy: RemovalPolicy.DESTROY
      }
    };

    const hitcounter = new LambdaToDynamoDB(this, 'LambdaToDynamoDB',
 lambda_ddb_props);

    // grant the hitcounter lambda role invoke permissions to the hello function
    helloFunc.grantInvoke(hitcounter.lambdaFunction);

    const api_lambda_props: ApiGatewayToLambdaProps = {
      existingLambdaObj: hitcounter.lambdaFunction,
      apiGatewayProps: {
        defaultMethodOptions: {
          authorizationType: api.AuthorizationType.NONE
        }
      }
    };

    new ApiGatewayToLambda(this, 'ApiGatewayToLambda', api_lambda_props);
  }
}
```

Python

Edit the file `hello_constructs/hello_constructs_stack.py` with the following:

```
from constructs import Construct
from aws_cdk import (
    aws_lambda as _lambda,
    aws_apigateway as apigw,
    aws_dynamodb as ddb,
    App,
    Stack,
    RemovalPolicy
)

from aws_solutions_constructs import (
    aws_apigateway_lambda as apigw_lambda,
```

```
    aws_lambda_dynamodb as lambda_ddb
)

class HelloConstructsStack(Stack):

    def __init__(self, scope: Construct, id: str, **kwargs) -> None:
        super().__init__(scope, id, **kwargs)

        # The code that defines your stack goes here

        self.hello_func = _lambda.Function(
          self, 'HelloHandler',
          runtime=_lambda.Runtime.PYTHON_3_7,
          handler='hello.handler',
          code=_lambda.Code.from_asset('lambda'),
        )

        #  hit counter, aws-lambda-dynamodb pattern
        self.hit_counter = lambda_ddb.LambdaToDynamoDB(
            self, 'LambdaToDynamoDB',
            lambda_function_props=_lambda.FunctionProps(
                runtime=_lambda.Runtime.PYTHON_3_7,
                code=_lambda.Code.from_asset('lambda'),
                handler='hitcounter.handler',
                environment={
                    'DOWNSTREAM_FUNCTION_NAME': self.hello_func.function_name
                }
            ),
            dynamo_table_props=ddb.TableProps(
                table_name='SolutionsConstructsHits',
                partition_key={
                    'name': 'path',
                    'type': ddb.AttributeType.STRING
                },
                removal_policy=RemovalPolicy.DESTROY
            )
        )

        # grant the hitcounter lambda role invoke permissions to the hello function
        self.hello_func.grant_invoke(self.hit_counter.lambda_function)

        apigw_lambda.ApiGatewayToLambda(
            self, 'ApiGatewayToLambda',
            existing_lambda_obj=self.hit_counter.lambda_function,
```

```
            api_gateway_props=apigw.RestApiProps(
                default_method_options=apigw.MethodOptions(
                    authorization_type=apigw.AuthorizationType.NONE
                )
            )
        )
```

Java

Edit the file `src/../HelloConstructsStack.java` with the following:

```
package com.myorg;

import java.util.Map;
import java.util.HashMap;

import software.constructs.Construct;
import software.amazon.awscdk.RemovalPolicy;
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;

import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awscdk.services.apigateway.*;
import software.amazon.awscdk.services.dynamodb.*;
import software.amazon.awsconstructs.services.apigatewaylambda.ApiGatewayToLambda;
import
 software.amazon.awsconstructs.services.apigatewaylambda.ApiGatewayToLambdaProps;
import software.amazon.awsconstructs.services.lambdadynamodb.*;

public class HelloConstructsStack extends Stack {
    public HelloConstructsStack(final Construct scope, final String id) {
        this(scope, id, null);
    }

    public HelloConstructsStack(final Construct scope, final String id, final
 StackProps props) {
        super(scope, id, props);

        final Function helloFunc = Function.Builder.create(this, "HelloHandler")
                .runtime(Runtime.NODEJS_14_X) // execution environment
                .code(Code.fromAsset("lambda")) // code loaded from the "lambda"
 directory
```

```
                .handler("hello.handler") // file is "hello", function is "handler"
                .build();

        final Map<String, String> lambdaEnvironment = new HashMap<>();
        lambdaEnvironment.put("DOWNSTREAM_FUNCTION_NAME",
 helloFunc.getFunctionName());

        final LambdaToDynamoDB hitcounter = new LambdaToDynamoDB(this,
 "LambdaToDynamoDBPattern",
                new LambdaToDynamoDBProps.Builder()
                        .lambdaFunctionProps(new FunctionProps.Builder()
                                .runtime(Runtime.NODEJS_14_X) // execution
 environment
                                .code(Code.fromAsset("lambda")) // code loaded from
 the "lambda" directory
                                .handler("hitcounter.handler") // file is
 "hitcounter", function is "handler"
                                .environment(lambdaEnvironment)
                                .build())
                        .dynamoTableProps(new TableProps.Builder()
                                .tableName("SolutionsConstructsHits")
                                .partitionKey(new Attribute.Builder()
                                        .name("path")
                                        .type(AttributeType.STRING)
                                        .build())
                                .removalPolicy(RemovalPolicy.DESTROY)
                                .build())
                        .build());

        // grant the hitcounter lambda role invoke permissions to the hello function
        helloFunc.grantInvoke(hitcounter.getLambdaFunction());

        final ApiGatewayToLambda apigwLambda = new ApiGatewayToLambda(this,
 "ApiGatewayToLambdaPattern",
                new ApiGatewayToLambdaProps.Builder()
                        .apiGatewayProps(new RestApiProps.Builder()
                                .defaultMethodOptions(MethodOptions.builder()
                                        .authorizationType(AuthorizationType.NONE)
                                        .build())
                                .build())
                        .existingLambdaObj(hitcounter.getLambdaFunction())
                        .build());
    }
}
```

Review the changes

Let's build our project and review the changes to our resources that will happen when we deploy this:

TypeScript

```
npm run build
cdk diff
```

Python

```
cdk diff
```

Java

```
mvn package
cdk diff
```

Our output should look like this:

```
IAM Statement Changes
####################################################################
#   # Resource   # Effect # Action     # Principal  # Condition    #
####################################################################
# - # ${ApiGatew # Allow  # lambda:Inv # Service:ap # "ArnLike": { #
#   # ayToLambda #        # okeFunctio # igateway.a #   "AWS:Sourc #
#   # PatternLam #        # n          # mazonaws.c # eArn": "arn: #
#   # bdaFunctio #        #            # om         # ${AWS::Parti #
#   # n5DC51B7E. #        #            #            # tion}:execut #
#   # Arn}       #        #            #            # e-api:${AWS: #
#   #            #        #            #            # :Region}:${A #
#   #            #        #            #            # WS::AccountI #
#   #            #        #            #            # d}:${ApiGate #
```

```
#   #            #        #            #            # wayToLambdaP #
#   #            #        #            #            # atternLambda #
#   #            #        #            #            # RestApiC0598 #
#   #            #        #            #            # E46}/${ApiGa #
#   #            #        #            #            # tewayToLambd #
#   #            #        #            #            # aPattern/Lam #
#   #            #        #            #            # bdaRestApi/D #
#   #            #        #            #            # eploymentSta #
#   #            #        #            #            # ge.prod}/*/* #
#   #            #        #            #            # "            #
#   #            #        #            #            # }            #
# - # ${ApiGatew # Allow  # lambda:Inv # Service:ap # "ArnLike": { #
#   # ayToLambda #        # okeFunctio # igateway.a #   "AWS:Sourc #
#   # PatternLam #        # n          # mazonaws.c # eArn": "arn: #
#   # bdaFunctio #        #            # om         # ${AWS::Parti #
#   # n5DC51B7E. #        #            #            # tion}:execut #
#   # Arn}       #        #            #            # e-api:${AWS: #
#   #            #        #            #            # :Region}:${A #
#   #            #        #            #            # WS::AccountI #
#   #            #        #            #            # d}:${ApiGate #
#   #            #        #            #            # wayToLambdaP #
#   #            #        #            #            # atternLambda #
#   #            #        #            #            # RestApiC0598 #
#   #            #        #            #            # E46}/test-in #
#   #            #        #            #            # voke-stage/* #
#   #            #        #            #            # /*"          #
#   #            #        #            #            # }            #
# - # ${ApiGatew # Allow  # lambda:Inv # Service:ap # "ArnLike": { #
#   # ayToLambda #        # okeFunctio # igateway.a #   "AWS:Sourc #
#   # PatternLam #        # n          # mazonaws.c # eArn": "arn: #
#   # bdaFunctio #        #            # om         # ${AWS::Parti #
#   # n5DC51B7E. #        #            #            # tion}:execut #
#   # Arn}       #        #            #            # e-api:${AWS: #
#   #            #        #            #            # :Region}:${A #
#   #            #        #            #            # WS::AccountI #
#   #            #        #            #            # d}:${ApiGate #
#   #            #        #            #            # wayToLambdaP #
#   #            #        #            #            # atternLambda #
#   #            #        #            #            # RestApiC0598 #
#   #            #        #            #            # E46}/${ApiGa #
#   #            #        #            #            # tewayToLambd #
#   #            #        #            #            # aPattern/Lam #
#   #            #        #            #            # bdaRestApi/D #
#   #            #        #            #            # eploymentSta #
```

```
#   #            #        #            #            # ge.prod}/*/" #
#   #            #        #            #            # }            #
# - # ${ApiGatew # Allow  # lambda:Inv # Service:ap # "ArnLike": { #
#   # ayToLambda #        # okeFunctio # igateway.a #   "AWS:Sourc #
#   # PatternLam #        # n          # mazonaws.c # eArn": "arn: #
#   # bdaFunctio #        #            # om         # ${AWS::Parti #
#   # n5DC51B7E. #        #            #            # tion}:execut #
#   # Arn}       #        #            #            # e-api:${AWS: #
#   #            #        #            #            # :Region}:${A #
#   #            #        #            #            # WS::AccountI #
#   #            #        #            #            # d}:${ApiGate #
#   #            #        #            #            # wayToLambdaP #
#   #            #        #            #            # atternLambda #
#   #            #        #            #            # RestApiC0598 #
#   #            #        #            #            # E46}/test-in #
#   #            #        #            #            # voke-stage/* #
#   #            #        #            #            # /"           #
#   #            #        #            #            # }            #
####################################################################
# - # *          # Allow  # xray:PutTe # AWS:${ApiG #              #
#   #            #        # lemetryRec # atewayToLa #              #
#   #            #        # ords       # mbdaPatter #              #
#   #            #        # xray:PutTr # nLambdaFun #              #
#   #            #        # aceSegment # ctionServi #              #
#   #            #        # s          # ceRole0C12 #              #
#   #            #        #            # 3D8D}      #              #
####################################################################
# + # ${HelloHan # Allow  # lambda:Inv # AWS:${Lamb #              #
#   # dler.Arn}  #        # okeFunctio # daToDynamo #              #
#   # ${HelloHan #        # n          # DBPattern/ #              #
#   # dler.Arn}: #        #            # LambdaFunc #              #
#   # *          #        #            # tionServic #              #
#   #            #        #            # eRole}     #              #
####################################################################
# + # ${HelloHan # Allow  # sts:Assume # Service:la #              #
#   # dler/Servi #        # Role       # mbda.amazo #              #
#   # ceRole.Arn #        #            # naws.com   #              #
#   # }          #        #            #            #              #
####################################################################
# + # ${LambdaTo # Allow  # dynamodb:B # AWS:${Lamb #              #
#   # DynamoDBPa #        # atchGetIte # daToDynamo #              #
#   # ttern/Dyna #        # m          # DBPattern/ #              #
#   # moTable.Ar #        # dynamodb:B # LambdaFunc #              #
#   # n}         #        # atchWriteI # tionServic #              #
```

```
#   #            #        # tem        # eRole}     #              #
#   #            #        # dynamodb:C #            #              #
#   #            #        # onditionCh #            #              #
#   #            #        # eckItem    #            #              #
#   #            #        # dynamodb:D #            #              #
#   #            #        # eleteItem  #            #              #
#   #            #        # dynamodb:D #            #              #
#   #            #        # escribeTab #            #              #
#   #            #        # le         #            #              #
#   #            #        # dynamodb:G #            #              #
#   #            #        # etItem     #            #              #
#   #            #        # dynamodb:G #            #              #
#   #            #        # etRecords  #            #              #
#   #            #        # dynamodb:G #            #              #
#   #            #        # etShardIte #            #              #
#   #            #        # rator      #            #              #
#   #            #        # dynamodb:P #            #              #
#   #            #        # utItem     #            #              #
#   #            #        # dynamodb:Q #            #              #
#   #            #        # uery       #            #              #
#   #            #        # dynamodb:S #            #              #
#   #            #        # can        #            #              #
#   #            #        # dynamodb:U #            #              #
#   #            #        # pdateItem  #            #              #
###################################################################
# + # ${LambdaTo # Allow  # lambda:Inv # Service:ap # "ArnLike": { #
#   # DynamoDBPa #        # okeFunctio # igateway.a #   "AWS:Sourc #
#   # ttern/Lamb #        # n          # mazonaws.c # eArn": "arn: #
#   # daFunction #        #            # om         # ${AWS::Parti #
#   # .Arn}      #        #            #            # tion}:execut #
#   #            #        #            #            # e-api:${AWS: #
#   #            #        #            #            # :Region}:${A #
#   #            #        #            #            # WS::AccountI #
#   #            #        #            #            # d}:${ApiGate #
#   #            #        #            #            # wayToLambdaP #
#   #            #        #            #            # atternLambda #
#   #            #        #            #            # RestApiC0598 #
#   #            #        #            #            # E46}/${ApiGa #
#   #            #        #            #            # tewayToLambd #
#   #            #        #            #            # aPattern/Lam #
#   #            #        #            #            # bdaRestApi/D #
#   #            #        #            #            # eploymentSta #
#   #            #        #            #            # ge.prod}/*/* #
#   #            #        #            #            # "            #
```

```
#   #            #        #            #            # }           #
# + # ${LambdaTo # Allow  # lambda:Inv # Service:ap # "ArnLike": { #
#   # DynamoDBPa #        # okeFunctio # igateway.a #   "AWS:Sourc #
#   # ttern/Lamb #        # n          # mazonaws.c # eArn": "arn: #
#   # daFunction #        #            # om         # ${AWS::Parti #
#   # .Arn}      #        #            #            # tion}:execut #
#   #            #        #            #            # e-api:${AWS: #
#   #            #        #            #            # :Region}:${A #
#   #            #        #            #            # WS::AccountI #
#   #            #        #            #            # d}:${ApiGate #
#   #            #        #            #            # wayToLambdaP #
#   #            #        #            #            # atternLambda #
#   #            #        #            #            # RestApiC0598 #
#   #            #        #            #            # E46}/test-in #
#   #            #        #            #            # voke-stage/* #
#   #            #        #            #            # /*"          #
#   #            #        #            #            # }            #
# + # ${LambdaTo # Allow  # lambda:Inv # Service:ap # "ArnLike": { #
#   # DynamoDBPa #        # okeFunctio # igateway.a #   "AWS:Sourc #
#   # ttern/Lamb #        # n          # mazonaws.c # eArn": "arn: #
#   # daFunction #        #            # om         # ${AWS::Parti #
#   # .Arn}      #        #            #            # tion}:execut #
#   #            #        #            #            # e-api:${AWS: #
#   #            #        #            #            # :Region}:${A #
#   #            #        #            #            # WS::AccountI #
#   #            #        #            #            # d}:${ApiGate #
#   #            #        #            #            # wayToLambdaP #
#   #            #        #            #            # atternLambda #
#   #            #        #            #            # RestApiC0598 #
#   #            #        #            #            # E46}/${ApiGa #
#   #            #        #            #            # tewayToLambd #
#   #            #        #            #            # aPattern/Lam #
#   #            #        #            #            # bdaRestApi/D #
#   #            #        #            #            # eploymentSta #
#   #            #        #            #            # ge.prod}/*/" #
#   #            #        #            #            # }            #
# + # ${LambdaTo # Allow  # lambda:Inv # Service:ap # "ArnLike": { #
#   # DynamoDBPa #        # okeFunctio # igateway.a #   "AWS:Sourc #
#   # ttern/Lamb #        # n          # mazonaws.c # eArn": "arn: #
#   # daFunction #        #            # om         # ${AWS::Parti #
#   # .Arn}      #        #            #            # tion}:execut #
#   #            #        #            #            # e-api:${AWS: #
#   #            #        #            #            # :Region}:${A #
#   #            #        #            #            # WS::AccountI #
```

```
#   #            #        #            #            # d}:${ApiGate #
#   #            #        #            #            # wayToLambdaP #
#   #            #        #            #            # atternLambda #
#   #            #        #            #            # RestApiC0598 #
#   #            #        #            #            # E46}/test-in #
#   #            #        #            #            # voke-stage/* #
#   #            #        #            #            # /"           #
#   #            #        #            #            # }            #
####################################################################
# + # ${LambdaTo # Allow  # sts:Assume # Service:la #              #
#   # DynamoDBPa #        # Role       # mbda.amazo #              #
#   # ttern/Lamb #        #            # naws.com   #              #
#   # daFunction #        #            #            #              #
#   # ServiceRol #        #            #            #              #
#   # e.Arn}     #        #            #            #              #
####################################################################
# + # *          # Allow  # xray:PutTe # AWS:${Lamb #              #
#   #            #        # lemetryRec # daToDynamo #              #
#   #            #        # ords       # DBPattern/ #              #
#   #            #        # xray:PutTr # LambdaFunc #              #
#   #            #        # aceSegment # tionServic #              #
#   #            #        # s          # eRole}     #              #
####################################################################
# + # arn:${AWS: # Allow  # logs:Creat # AWS:${Lamb #              #
#   # :Partition #        # eLogGroup  # daToDynamo #              #
#   # }:logs:${A #        # logs:Creat # DBPattern/ #              #
#   # WS::Region #        # eLogStream # LambdaFunc #              #
#   # }:${AWS::A #        # logs:PutLo # tionServic #              #
#   # ccountId}: #        # gEvents    # eRole}     #              #
#   # log-group: #        #            #            #              #
#   # /aws/lambd #        #            #            #              #
#   # a/*        #        #            #            #              #
####################################################################
IAM Policy Changes
####################################################################
#   # Resource                     # Managed Policy ARN            #
####################################################################
# + # ${HelloHandler/ServiceRole}  # arn:${AWS::Partition}:iam::aw #
#   #                              # s:policy/service-role/AWSLamb #
#   #                              # daBasicExecutionRole          #
####################################################################
(NOTE: There may be security-related changes not in this list. See https://github.com/
aws/aws-cdk/issues/1299)
```

```
Resources
[-] AWS::IAM::Role ApiGatewayToLambdaPatternLambdaFunctionServiceRole0C123D8D destroy
[-] AWS::IAM::Policy
 ApiGatewayToLambdaPatternLambdaFunctionServiceRoleDefaultPolicy253751F2 destroy
[-] AWS::Lambda::Function ApiGatewayToLambdaPatternLambdaFunction5DC51B7E destroy
[-] AWS::ApiGateway::Deployment
 ApiGatewayToLambdaPatternLambdaRestApiDeployment4109DB93346ab5d96a64d161f4cf4f020d3cdf94
 destroy
[+] AWS::IAM::Role HelloHandler/ServiceRole HelloHandlerServiceRole11EF7C63
[+] AWS::Lambda::Function HelloHandler HelloHandler2E4FBA4D
[+] AWS::IAM::Role LambdaToDynamoDBPattern/LambdaFunctionServiceRole
 LambdaToDynamoDBPatternLambdaFunctionServiceRoleAAE562DF
[+] AWS::IAM::Policy LambdaToDynamoDBPattern/LambdaFunctionServiceRole/DefaultPolicy
 LambdaToDynamoDBPatternLambdaFunctionServiceRoleDefaultPolicy13FCEF5D
[+] AWS::Lambda::Function LambdaToDynamoDBPattern/LambdaFunction
 LambdaToDynamoDBPatternLambdaFunctionCEB12909
[+] AWS::DynamoDB::Table LambdaToDynamoDBPattern/DynamoTable
 LambdaToDynamoDBPatternDynamoTable4B679F88
[+] AWS::ApiGateway::Deployment ApiGatewayToLambdaPattern/LambdaRestApi/Deployment
 ApiGatewayToLambdaPatternLambdaRestApiDeployment4109DB93cb22f0825c29c1c2a437398f022ebdb0
[~] AWS::ApiGateway::Stage ApiGatewayToLambdaPattern/LambdaRestApi/DeploymentStage.prod
 ApiGatewayToLambdaPatternLambdaRestApiDeploymentStageprodFDEB8074
 ## [~] DeploymentId
     ## [~] .Ref:
         ## [-]
 ApiGatewayToLambdaPatternLambdaRestApiDeployment4109DB93346ab5d96a64d161f4cf4f020d3cdf94
         ## [+]
 ApiGatewayToLambdaPatternLambdaRestApiDeployment4109DB93cb22f0825c29c1c2a437398f022ebdb0
[~] AWS::Lambda::Permission ApiGatewayToLambdaPattern/LambdaRestApi/Default/{proxy+}/
ANY/
ApiPermission.HelloConstructsStackApiGatewayToLambdaPatternLambdaRestApi553584F5.ANY..
{proxy+}
 ApiGatewayToLambdaPatternLambdaRestApiproxyANYApiPermissionHelloConstructsStackApiGatewayToLam
 replace
 ## [~] FunctionName (requires replacement)
     ## [~] .Fn::GetAtt:
         ## @@ -1,4 +1,4 @@
            [ ] [
            [-]   "ApiGatewayToLambdaPatternLambdaFunction5DC51B7E",
            [+]   "LambdaToDynamoDBPatternLambdaFunctionCEB12909",
            [ ]   "Arn"
            [ ] ]
[~] AWS::Lambda::Permission ApiGatewayToLambdaPattern/LambdaRestApi/Default/{proxy+}/
ANY/
```

```
ApiPermission.Test.HelloConstructsStackApiGatewayToLambdaPatternLambdaRestApi553584F5.ANY..
{proxy+}
 ApiGatewayToLambdaPatternLambdaRestApiproxyANYApiPermissionTestHelloConstructsStackApiGatewayT
 replace
 ## [~] FunctionName (requires replacement)
     ## [~] .Fn::GetAtt:
         ## @@ -1,4 +1,4 @@
            [ ] [
            [-]   "ApiGatewayToLambdaPatternLambdaFunction5DC51B7E",
            [+]   "LambdaToDynamoDBPatternLambdaFunctionCEB12909",
            [ ]   "Arn"
            [ ] ]
[~] AWS::ApiGateway::Method ApiGatewayToLambdaPattern/LambdaRestApi/Default/{proxy+}/
ANY ApiGatewayToLambdaPatternLambdaRestApiproxyANY321FD2C2
 ## [~] Integration
     ## [~] .Uri:
         ## [~] .Fn::Join:
             ## @@ -12,7 +12,7 @@
                [ ] ":lambda:path/2015-03-31/functions/",
                [ ] {
                [ ]   "Fn::GetAtt": [
                [-]     "ApiGatewayToLambdaPatternLambdaFunction5DC51B7E",
                [+]     "LambdaToDynamoDBPatternLambdaFunctionCEB12909",
                [ ]     "Arn"
                [ ]   ]
                [ ] },
[~] AWS::Lambda::Permission ApiGatewayToLambdaPattern/LambdaRestApi/Default/ANY/
ApiPermission.HelloConstructsStackApiGatewayToLambdaPatternLambdaRestApi553584F5.ANY..
 ApiGatewayToLambdaPatternLambdaRestApiANYApiPermissionHelloConstructsStackApiGatewayToLambdaPa
 replace
 ## [~] FunctionName (requires replacement)
     ## [~] .Fn::GetAtt:
         ## @@ -1,4 +1,4 @@
            [ ] [
            [-]   "ApiGatewayToLambdaPatternLambdaFunction5DC51B7E",
            [+]   "LambdaToDynamoDBPatternLambdaFunctionCEB12909",
            [ ]   "Arn"
            [ ] ]
[~] AWS::Lambda::Permission ApiGatewayToLambdaPattern/LambdaRestApi/Default/ANY/
ApiPermission.Test.HelloConstructsStackApiGatewayToLambdaPatternLambdaRestApi553584F5.ANY..
 ApiGatewayToLambdaPatternLambdaRestApiANYApiPermissionTestHelloConstructsStackApiGatewayToLamb
 replace
 ## [~] FunctionName (requires replacement)
     ## [~] .Fn::GetAtt:
```

```
          ## @@ -1,4 +1,4 @@
             [ ] [
             [-]   "ApiGatewayToLambdaPatternLambdaFunction5DC51B7E",
             [+]   "LambdaToDynamoDBPatternLambdaFunctionCEB12909",
             [ ]   "Arn"
             [ ] ]
[~] AWS::ApiGateway::Method ApiGatewayToLambdaPattern/LambdaRestApi/Default/ANY
 ApiGatewayToLambdaPatternLambdaRestApiANY07ADEFED
 ## [~] Integration
     ## [~] .Uri:
         ## [~] .Fn::Join:
             ## @@ -12,7 +12,7 @@
                 [ ] ":lambda:path/2015-03-31/functions/",
                 [ ] {
                 [ ]   "Fn::GetAtt": [
                 [-]     "ApiGatewayToLambdaPatternLambdaFunction5DC51B7E",
                 [+]     "LambdaToDynamoDBPatternLambdaFunctionCEB12909",
                 [ ]     "Arn"
                 [ ]   ]
                 [ ] },
```

cdk deploy

Okay, ready to deploy?

```
cdk deploy
```

Stack outputs

When deployment is complete, you'll notice this line:

```
Outputs:
  HelloConstructsStack.RestApiEndpoint0551178A = https://
      xxxxxxxxxx
      .execute-api.us-east-1.amazonaws.com/prod/
```

Testing your app

Let’s try to hit this endpoint with curl. Copy the URL and execute (your prefix and region will likely be different).

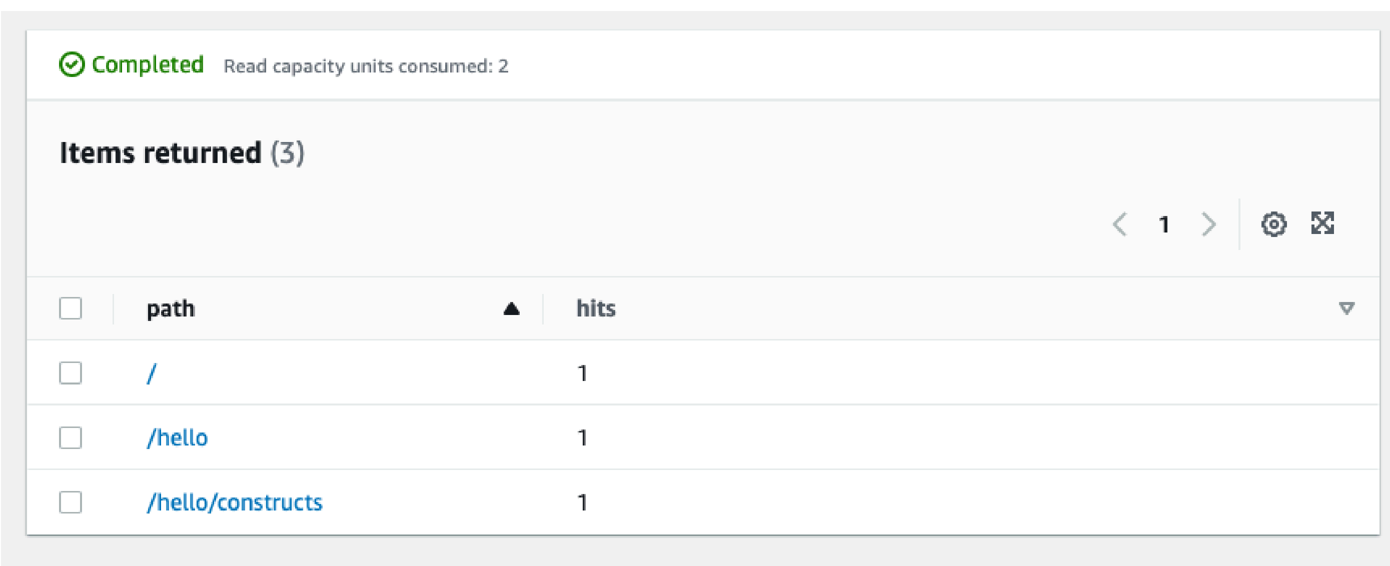
```
curl https://
  xxxxxxxxxx
  .execute-api.us-east-1.amazonaws.com/prod/
```

Output should look like this:

```
Hello, AWS Solutions Constructs! You've hit /
```

Now, let's review the `SolutionsConstructsHits` Amazon DynamoDB table.

1. Go to the DynamoDB console.
2. Make sure you are in the Region where you created the table.
3. Select **Tables** in the navigation pane and select the **SolutionsConstructsHits** table.
4. Open the table and select “Items” or “Explore table items” (depending upon which version of the Dyamodb console you are using).
5. You should see how many hits you got for each path (the data below reflects running the curl command 3 times with 3 different resources at the end of the URL).



6. Try hitting a new path and refresh the Items view. You should see a new item with a `hits` count of one.

If this is the output you received, your app works!

Clean up

To avoid unexpected charges to your account, make sure you clean up your CDK stack.

You can either delete the stack through the AWS CloudFormation console or use cdk destroy:

```
cdk destroy
```

You'll be asked:

```
Are you sure you want to delete: HelloConstructsStack (y/n)?
```

Hit “y” and you’ll see your stack being destroyed.

Sample Use Cases

This library includes a collection of functional use case implementations to demonstrate the usage of Constructs architectural patterns. These can be used in the same way as architectural patterns, and can be conceptualized as an additional "higher-level" abstraction of those patterns. The following use cases are provided as functional examples.

AWS Glue Custom ETL Job

This use case implements an example using the Kinesis Data Streams Glue Job construct. The architecture uses a custom ETL job defined in AWS Glue that takes in data from Amazon Kinesis Data Streams to process and store it in the target datastore as defined by the ETL script (for this example, an S3 bucket location).

Source Code (aws-custom-glue-etl)

https://github.com/awslabs/aws-solutions-constructs/tree/master/source/use_cases/aws-custom-glue-etl

AWS Static S3 Website

This use case pattern (`aws-s3-static-website`) implements an Amazon CloudFront distribution, Amazon S3 bucket, and AWS Lambda-based custom resource to copy the static website content for the Wild Rydes demo website (part of the `aws-serverless-web-app` implementation).

Source Code (aws-s3-static-website)

https://github.com/awslabs/aws-solutions-constructs/tree/master/source/use_cases/aws-s3-static-website

AWS Restaurant Management Demo

This use case pattern (`aws-restaurant-management-demo`) implements a complex, multi-stack architecture that models a restaurant management system. This use case will provision a stack for service/wait staff to open/close orders, a stack for kitchen staff to view/complete orders, and a stack for managers to perform various business functions. It will also provision a stack containing a central DynamoDB table for managing orders, as well as a Lambda layer for sharing common database access patterns.

Source Code (aws-restaurant-management-demo)

https://github.com/awslabs/aws-solutions-constructs/tree/master/source/use_cases/aws-restaurant-management-demo

API Reference

AWS Solutions Constructs (Constructs) is an open-source extension of the AWS Cloud Development Kit (AWS CDK) that provides multi-service, well-architected patterns for quickly defining solutions in code to create predictable and repeatable infrastructure. Constructs's goal is to accelerates the experience for developers to build solutions of any size using pattern-based definitions for their architecture.

The patterns defined in Constructs are high level, multi-service abstractions of AWS CDK constructs that have default configurations based on well-architected best practices. The library is organized into logical modules using object-oriented techniques to create each architectural pattern model.

The CDK is available in the following languages:

- JavaScript, TypeScript (Node.js ≥ 10.3.0)
- Python (Python ≥ 3.6)
- Java (Java ≥ 1.8)

Modules

AWS Solutions Constructs is organized into several modules. They are named like this:

- **aws-xxx**: Well-architected pattern package for the indicated services. This package will contain constructs that contain multiple AWS CDK service modules to configure the given pattern.
- **xxx**: Packages that don't start "**aws-**" are Constructs core modules that are used to configure best practice defaults for services used within the pattern library.

Module Contents

Modules contain the following types:

- **Patterns** - All higher-level, multi-services constructs in this library.
- **Other Types** - All non-construct classes, interfaces, structs and enums that exist to support the patterns.

Patterns take a set of (input) properties in their constructor; the set of properties (and which ones are required) can be seen on a pattern's documentation page.

The pattern's documentation page also lists the available methods to call and the properties which can be used to retrieve information about the pattern after it has been instantiated.

aws-alb-fargate

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_alb_fargate`
Typescript	`@aws-solutions-constructs/aws-alb-fargate`
Java	`software.amazon.awsconstructs.services.albfargate`

Overview

This AWS Solutions Construct implements an an Application Load Balancer to an AWS Fargate service

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { AlbToFargate, AlbToFargateProps } from '@aws-solutions-constructs/aws-alb-fargate';
import * as acm from 'aws-cdk-lib/aws-certificatemanager';
```

```
const certificate = acm.Certificate.fromCertificateArn(
    this,
    'existing-cert',
    "arn:aws:acm:us-
east-1:123456789012:certificate/11112222-3333-1234-1234-123456789012"
);

const constructProps: AlbToFargateProps = {
    ecrRepositoryArn: "arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-repo",
    ecrImageVersion: "latest",
    listenerProps: {
        certificates: [certificate]
    },
    publicApi: true
};

// Note - all alb constructs turn on ELB logging by default, so require that an
 environment including account
// and region be provided when creating the stack
//
// new MyStack(app, 'id', {env: {account: '123456789012', region: 'us-east-1' }});
new AlbToFargate(this, 'new-construct', constructProps);
```

Python

```
from aws_solutions_constructs.aws_alb_fargate import AlbToFargate, AlbToFargateProps
from aws_cdk import (
    aws_certificatemanager as acm,
    aws_elasticloadbalancingv2 as alb,
    Stack
)
from constructs import Construct

# Obtain a pre-existing certificate from your account
certificate = acm.Certificate.from_certificate_arn(
      self,
      'existing-cert',
      "arn:aws:acm:us-
east-1:123456789012:certificate/11112222-3333-1234-1234-123456789012"
    )
```

```
# Note - all alb constructs turn on ELB logging by default, so require that an
 environment including account
# and region be provided when creating the stack
#
# MyStack(app, 'id', env=cdk.Environment(account='123456789012', region='us-
east-1'))
AlbToFargate(self, 'new-construct',
                ecr_repository_arn="arn:aws:ecr:us-east-1:123456789012:repository/
your-ecr-repo",
                ecr_image_version="latest",
                listener_props=alb.BaseApplicationListenerProps(
                    certificates=[certificate],
                ),
                public_api=True)
```

Java

```
import software.constructs.Construct;
import java.util.List;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.elasticloadbalancingv2.*;
import software.amazon.awsconstructs.services.albfargate.*;

// The code that defines your stack goes here
// Obtain a pre-existing certificate from your account
ListenerCertificate listenerCertificate = ListenerCertificate
        .fromArn("arn:aws:acm:us-
east-1:123456789012:certificate/11112222-3333-1234-1234-123456789012");

// Note - all alb constructs turn on ELB logging by default, so require that an
 environment including account
// and region be provided when creating the stack
//
// new MyStack(app, "id", StackProps.builder()
//         .env(Environment.builder()
//                 .account("123456789012")
//                 .region("us-east-1")
//                 .build());
new AlbToFargate(this, "AlbToFargatePattern", new AlbToFargateProps.Builder()
```

```
        .ecrRepositoryArn("arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-
repo")
        .ecrImageVersion("latest")
        .listenerProps(new BaseApplicationListenerProps.Builder()
                .certificates(List.of(listenerCertificate))
                .build())
        .publicApi(true)
        .build());
```

Pattern Construct Props

Name	Type	Description
publicApi	boolean	Whether the construct is deploying a private or public API. This has implications for the VPC and ALB.
loadBalancerProps?	[elasticloadbalancingv2.ApplicationLoadBalancerProps]()	Optional custom properties for a new loadBalancer. Providing both this and existingLoadBalancer is an error. This cannot specify a VPC, it will use the VPC in existingVpc or the VPC created by the construct.
existingLoadBalancerObj?	[elasticloadbalancingv2.ApplicationLoadBalancer]()	[existing Application Load Balancer to incorporate into the construct architecture. Providing both this and loadBalancerProps is an error. The VPC containing this loadBalancer must match the VPC provided in existingVpc.
listenerProps?	[ApplicationListenerProps]()	Props to define the listener. Must be provided when adding

Name	Type	Description
		the listener to an ALB (eg - when creating the alb), may not be provided when adding a second target to an already established listener. When provided, must include either a certificate or protocol: HTTP
targetGroupProps?	ApplicationTargetGroupProps	Optional custom properties for a new target group. If your application requires end to end encryption, then you should set the protocol attribute to elb.ApplicationProtocol.HTTPS and use a container that can accept HTTPS traffic.
ruleProps?	AddRuleProps	Rules for directing traffic to the target being created. Must not be specified for the first listener added to an ALB, and must be specified for the second target added to a listener. Add a second target by instantiating this construct a second time and providing the existingAlb from the first instantiation.

Name	Type	Description
vpcProps?	[ec2.VpcProps]	Optional custom properties for a VPC the construct will create. This VPC will be used by the new ALB and any Private Hosted Zone the construct creates (that's why loadBalan cerProps and privateHo stedZoneProps can't include a VPC). Providing both this and existingVpc is an error.
existingVpc?	[ec2.IVpc]	An existing VPC in which to deploy the construct. Providing both this and vpcProps is an error. If the client provides an existing load balancer and/or existing Private Hosted Zone, those constructs must exist in this VPC.
logAlbAccessLogs?	boolean	Whether to turn on Access Logs for the Application Load Balancer. Uses an S3 bucket with associated storage costs.Enabling Access Logging is a best practice. default - true
albLoggingBucketProps?	[s3.BucketProps]	Optional properties to customize the bucket used to store the ALB Access Logs. Supplying this and setting logAccessLogs to false is an error. @default - none

Name	Type	Description
clusterProps?	ecs.ClusterProps	Optional properties to create a new ECS cluster. To provide an existing cluster, use the cluster attribute of fargateServiceProp s.
ecrRepositoryArn?	string	The arn of an ECR Repository containing the image to use to generate the containers. Either this or the image property of containerDefinitionProps must be provided. format: arn:aws:ecr:*region*:*account number*:repository/*Repository Name*
ecrImageVersion?	string	The version of the image to use from the repository. Defaults to "Latest"
containerDefinitionProps?	ecs.ContainerDefinitionProps | any	Optional props to define the container created for the Fargate Service (defaults found in fargate-defaults.ts)
fargateTaskDefinit ionProps?	ecs.FargateTaskDefinitionProps | any	Optional props to define the Fargate Task Definition for this construct (defaults found in fargate-defaults.ts)

Name	Type	Description
fargateServiceProps?	ecs.FargateServiceProps | any	Optional properties to override default values for the Fargate service. Service will set up in the Public or Isolated subnets of the VPC by default, override that (e.g. - choose Private subnets) by setting vpcSubnets on this object.
existingFargateSer viceObject?	ecs.FargateService	A Fargate Service already instantiated (probably by another Solutions Construct). If this is specified, then no props defining a new service can be provided, including: existingI mageObject, ecrImageVersion, containerDefinitionProps, fargateTaskDefinitionProps, ecrRepositoryArn, fargateSe rviceProps, clusterProps, existingClusterInterface
existingContainerD efinitionObject?	ecs.ContainerDefinition	A container definition already instantiated as part of a Fargate service. This must be the container in the existingF argateServiceObject

Pattern Properties

Name	Type	Description
vpc	ec2.IVpc	The VPC used by the construct (whether created by the

Name	Type	Description
		construct or providedb by the client)
loadBalancer	elasticloadbalancingv2.ApplicationLoadBalancer	The Load Balancer used by the construct (whether created by the construct or provided by the client)
listener	elb.ApplicationListener	The listener used by this pattern.
service	ecs.FargateService	The AWS Fargate service used by this construct (whether created by this construct or passed to this construct at initialization)
container	ecs.ContainerDefinition	The container associated with the AWS Fargate service in the service property.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Application Load Balancer

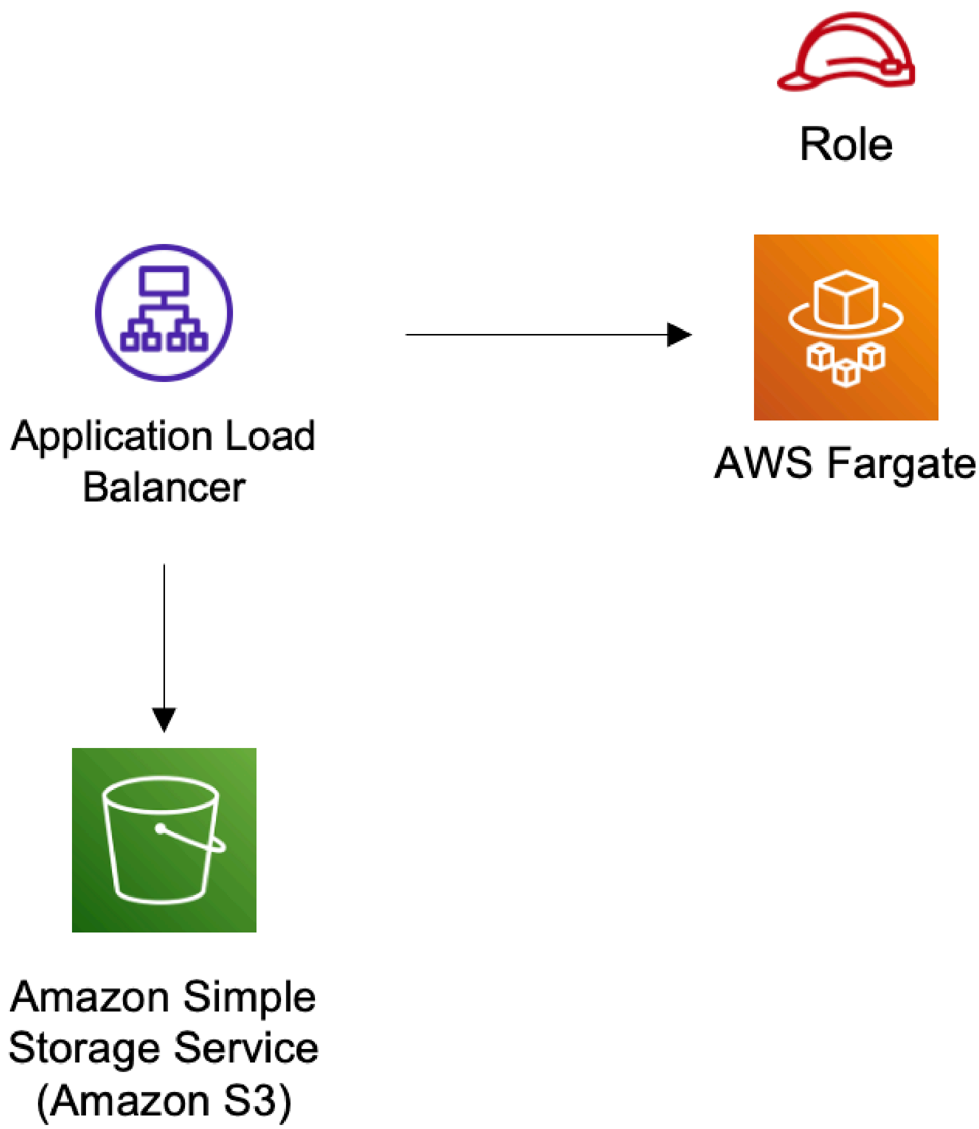
- Creates or configures an Application Load Balancer with:
  - Required listeners
  - New target group with routing rules if appropriate

AWS Fargate Service

- Sets up an AWS Fargate service as a target of the Application Load Balancer
  - Uses the existing service if provided

- Creates a new service if none provided.
  - Service will run in isolated subnets if available, then private subnets if available and finally public subnets

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-alb-fargate

aws-alb-lambda

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_alb_lambda`
Typescript	`@aws-solutions-constructs/aws-alb-lambda`
Java	`software.amazon.awsconstructs.services.alblambda`

Overview

This AWS Solutions Construct implements an an Application Load Balancer to an AWS Lambda function

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { AlbToLambda, AlbToLambdaProps } from '@aws-solutions-constructs/aws-alb-lambda';
import * as acm from 'aws-cdk-lib/aws-certificatemanager';
import * as lambda from 'aws-cdk-lib/aws-lambda';

// Obtain a pre-existing certificate from your account
const certificate = acm.Certificate.fromCertificateArn(
    this,
    'existing-cert',
    "arn:aws:acm:us-east-1:123456789012:certificate/11112222-3333-1234-1234-123456789012"
);

const constructProps: AlbToLambdaProps = {
  lambdaFunctionProps: {
    code: lambda.Code.fromAsset(`lambda`),
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler'
  },
  listenerProps: {
    certificates: [certificate]
  },
  publicApi: true
};

// Note - all alb constructs turn on ELB logging by default, so require that an
 environment including account
// and region be provided when creating the stack
//
// new MyStack(app, 'id', {env: {account: '123456789012', region: 'us-east-1' }});
new AlbToLambda(this, 'new-construct', constructProps);
```

Python

```
from aws_solutions_constructs.aws_alb_lambda import AlbToLambda, AlbToLambdaProps
from aws_cdk import (
    aws_certificatemanager as acm,
    aws_lambda as _lambda,
    aws_elasticloadbalancingv2 as alb,
    Stack
```

```
)
from constructs import Construct

# Obtain a pre-existing certificate from your account
certificate = acm.Certificate.from_certificate_arn(
  self,
  'existing-cert',
  "arn:aws:acm:us-
east-1:123456789012:certificate/11112222-3333-1234-1234-123456789012"
)

# Note - all alb constructs turn on ELB logging by default, so require that an
 environment including account
# and region be provided when creating the stack
#
# MyStack(app, 'id', env=cdk.Environment(account='123456789012', region='us-
east-1'))
AlbToLambda(self, 'new-construct',
            lambda_function_props=_lambda.FunctionProps(
                runtime=_lambda.Runtime.PYTHON_3_7,
                code=_lambda.Code.from_asset('lambda'),
                handler='index.handler',
            ),
            listener_props=alb.BaseApplicationListenerProps(
                certificates=[certificate]
            ),
            public_api=True)
```

Java

```
import software.constructs.Construct;
import java.util.List;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.elasticloadbalancingv2.*;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.alblambda.*;

// Obtain a pre-existing certificate from your account
ListenerCertificate listenerCertificate = ListenerCertificate
```

```
        .fromArn("arn:aws:acm:us-
east-1:123456789012:certificate/11112222-3333-1234-1234-123456789012");

// Note - all alb constructs turn on ELB logging by default, so require that an
 environment including account
// and region be provided when creating the stack
//
// new MyStack(app, "id", StackProps.builder()
//         .env(Environment.builder()
//                 .account("123456789012")
//                 .region("us-east-1")
//                 .build());
new AlbToLambda(this, "AlbToLambdaPattern", new AlbToLambdaProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .listenerProps(new BaseApplicationListenerProps.Builder()
                .certificates(List.of(listenerCertificate))
                .build())
        .publicApi(true)
        .build());
```

Pattern Construct Props

Name	Type	Description
loadBalancerProps?	elasticloadbalancingv2.ApplicationLoadBalancerProps	Optional custom properties for a new loadBalancer. Providing both this and existingLoadBalancer is an error. This cannot specify a VPC, it will use the VPC in existingVpc or the VPC created by the construct.
existingLoadBalancerObj?	elasticloadbalancingv2.ApplicationLoadBalancer	Existing Application Load Balancer to incorporate into the construct architect

Name	Type	Description
		ure. Providing both this and loadBalancerProps is an error. The VPC containing this loadBalancer must match the VPC provided in existingVpc.
listenerProps?	[ApplicationListenerProps]()	Props to define the listener. Must be provided when adding the listener to an ALB (eg - when creating the alb), may not be provided when adding a second target to an already established listener. When provided, must include either a certificate or protocol: HTTP
targetProps?	[ApplicationTargetGroupProps]()	Optional custom properties for a new target group. While this is a standard attribute of props for ALB constructs, there are few pertinent properties for a Lambda target.
ruleProps?	[AddRuleProps]()	Rules for directing traffic to the target being created. May not be specified for the first listener added to an ALB, and must be specified for the second target added to a listener. Add a second target by instantiating this construct a second time and providing the existingAlb from the first instantiation.

Name	Type	Description
vpcProps?	ec2.VpcProps	Optional custom properties for a VPC the construct will create. This VPC will be used by the new ALB and any Private Hosted Zone the construct creates (that's why loadBalancerProps and privateHostedZoneProps can't include a VPC). Providing both this and existingVpc is an error.
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	Optional user provided props to override the default props for the Lambda function.
existingVpc?	ec2.IVpc	An existing VPC in which to deploy the construct. Providing both this and vpcProps is an error. If the client provides an existing load balancer and/or existing Private Hosted Zone, those constructs must exist in this VPC.

Name	Type	Description
logAlbAccessLogs?	boolean	Whether to turn on Access Logs for the Application Load Balancer. Uses an S3 bucket with associated storage costs.Enabling Access Logging is a best practice. default - true
albLoggingBucketProps?	s3.BucketProps	Optional properties to customize the bucket used to store the ALB Access Logs. Supplying this and setting logAccessLogs to false is an error. @default - none
publicApi	boolean	Whether the construct is deploying a private or public API. This has implications for the VPC and ALB.

Pattern Properties

Name	Type	Description
vpc	ec2.IVpc	The VPC used by the construct (whether created by the construct or providedb by the client)
loadBalancer	elasticloadbalancingv2.ApplicationLoadBalancer	The Load Balancer used by the construct (whether created by the construct or provided by the client)

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of the Lambda function used in the pattern.
listener	`elb.ApplicationListener`	The listener used by this pattern.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

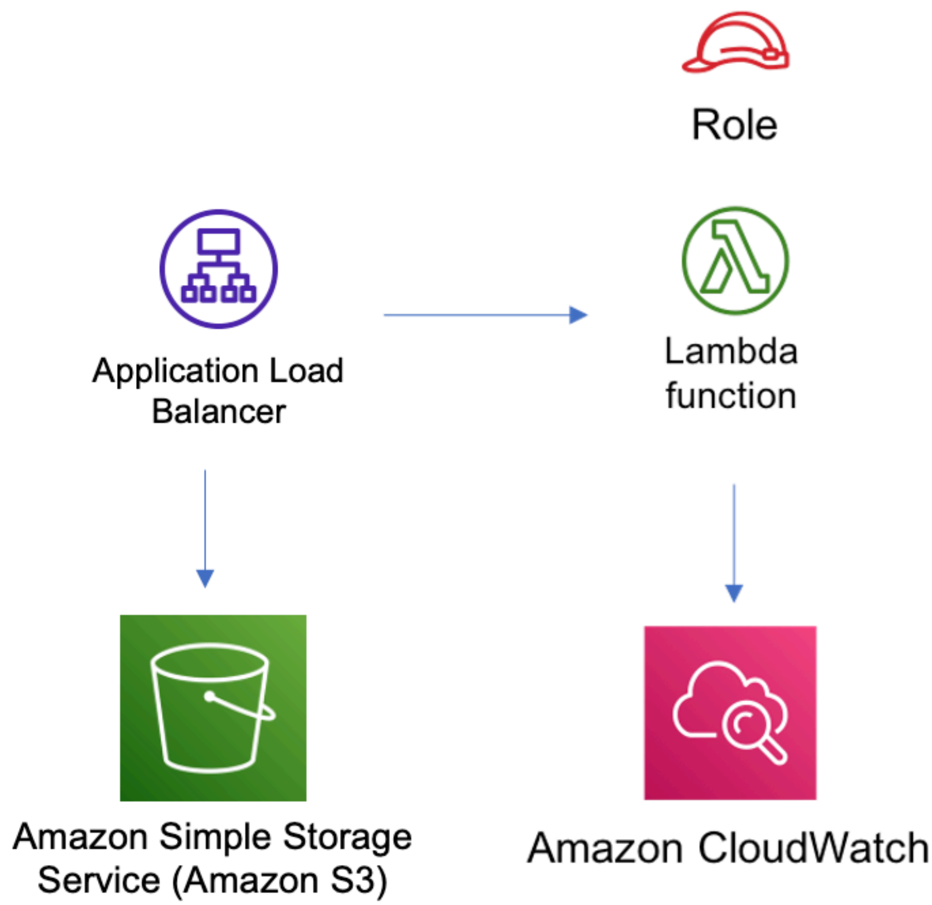
Application Load Balancer

- Creates or configures an Application Load Balancer with:
  - Required listeners
  - New target group with routing rules if appropriate

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-alb-lambda

aws-apigateway-dynamodb

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_apigateway_dynamodb
Typescript	@aws-solutions-constructs/aws-apigateway-dynamodb
Java	software.amazon.awsconstructs.services.apigatewaydynamodb

Overview

This AWS Solutions Construct implements an Amazon API Gateway REST API connected to Amazon DynamoDB table.

Here is a minimal deployable pattern definition in:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { ApiGatewayToDynamoDBProps, ApiGatewayToDynamoDB } from "@aws-solutions-constructs/aws-apigateway-dynamodb";

new ApiGatewayToDynamoDB(this, 'test-api-gateway-dynamodb-default', {});
```

Python

```
from aws_solutions_constructs.aws_apigateway_dynamodb import ApiGatewayToDynamoDB
from aws_cdk import Stack
from constructs import Construct

ApiGatewayToDynamoDB(self, 'test-api-gateway-dynamodb-default')
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.apigatewaydynamodb.*;

new ApiGatewayToDynamoDB(this, "test-api-gateway-dynamodb-default", new
 ApiGatewayToDynamoDBProps.Builder()
        .build());
```

Pattern Construct Props

Name	Type	Description
dynamoTableProps?	`dynamodb.TableProps`	Optional user provided props to override the default props for DynamoDB Table.
existingTableObj?	`dynamodb.Table`	Existing instance of DynamoDB table object, providing both this and `dynamoTableProps` will cause an error.
apiGatewayProps?	`api.RestApiProps`	Optional user-provided props to override the default props for the API Gateway.

Name	Type	Description
resourceName?	`string`	Optional name of the resource on the API Gateway. Defaults to the table's partitionKeyName
allowCreateOperation?	`boolean`	Whether to deploy an API Gateway Method for POST HTTP operations on the DynamoDB table (i.e. dynamodb:PutItem).
createRequestTemplate?	`string`	API Gateway Request Template for the create method for the default `application/json` content-type. This property is required if the `allowCreateOperation` property is set to true.
additionalCreateRequestTemplates?	`{ [contentType: string]: string; }`	Optional Create Request Templates for content-types other than `application/json`. Use the `createRequestTemplate` property to set the request template for the `application/json` content-type. This property can only be specified if the `allowCreateOperation` property is set to true.

Name	Type	Description
createIntegrationResponses?	api.IntegrationResponses[]	Optional, custom API Gateway Integration Response for the create method. This property can only be specified if the `allowCreateOperation` property is set to true.
allowReadOperation?	`boolean`	Whether to deploy an API Gateway Method for GET HTTP operations on DynamoDB table (i.e. dynamodb:Query).
readRequestTemplate?	`string`	API Gateway Request Template for the read method for the default `application/json` content-type. The default template only supports a partition key and not partition + sort keys.
additionalReadRequestTemplates?	`{ [contentType: string]: string; }`	Optional Read Request Templates for content-types other than `application/json`. Use the `readRequestTemplate` property to set the request template for the `application/json` content-type.
readIntegrationResponses?	api.IntegrationResponses[]	Optional, custom API Gateway Integration Response for the read method.

Name	Type	Description
allowUpdateOperation?	`boolean`	Whether to deploy API Gateway Method for PUT HTTP operations on DynamoDB table (i.e. dynamodb:UpdateItem).
updateRequestTemplate?	`string`	API Gateway Request Template for the update method. This property is required if the `allowUpdateOperation` property is set to true.
additionalUpdateRequestTemplates?	`{ [contentType: string]: string; }`	Optional Update Request Templates for content-types other than `application/json`. Use the `updateRequestTemplate` property to set the request template for the `application/json` content-type. This property can only be specified if the `allowUpdateOperation` property is set to true.
updateIntegrationResponses?	[`api.IntegrationResponses[]`]()	Optional, custom API Gateway Integration Response for the update method. This property can only be specified if the `allowUpdateOperation` property is set to true.
allowDeleteOperation?	`boolean`	Whether to deploy API Gateway Method for DELETE HTTP operations on DynamoDB table (i.e. dynamodb:DeleteItem).

Name	Type	Description
deleteRequestTemplate?	`string`	API Gateway Request Template for the delete method for the default `application/json` content-type.
additionalDeleteRequestTemplates?	`{ [contentType: string]: string; }`	Optional Delete request templates for content-types other than `application/json`. Use the `deleteRequestTemplate` property to set the request template for the `application/json` content-type. This property can only be specified if the `allowDeleteOperation` property is set to true.
deleteIntegrationResponses?	[`api.IntegrationResponses[]`]()	Optional, custom API Gateway Integration Response for the delete method. This property can only be specified if the `allowDeleteOperation` property is set to true.
logGroupProps?	[`logs.LogGroupProps`]()	User provided props to override the default props for for the CloudWatchLogs LogGroup.

Pattern Properties

Name	Type	Description
apiGateway	`api.RestApi`	Returns an instance of the api.RestApi created by the construct.
apiGatewayRole	`iam.Role`	Returns an instance of the iam.Role created by the construct for API Gateway.
dynamoTable	`dynamodb.Table`	Returns an instance of dynamodb.Table created by the construct.
apiGatewayCloudWatchRole?	`iam.Role`	Returns an instance of the iam.Role created by the construct for API Gateway for CloudWatch access.
apiGatewayLogGroup	`logs.LogGroup`	Returns an instance of the LogGroup created by the construct for API Gateway access logging to CloudWatch.

API Gateway Request/Response Template Properties Overview

This construct allows you to implement four DynamoDB API operations, CREATE/READ/UPDATE/DELETE (corresponding the HTTP POST/GET/PUT/DELETE requests respectively). They are completely independent and each follows the same pattern: * Setting `allowCreateOperation` to true will implement the `application/json` content-type with default request and response templates * The request template for `application/json` requests can be customized using the `createRequestTemplate` prop value * *Additional* request templates can be specified using the `additionalCreateRequestTemplates` prop value. Note - these DO NOT replace the `application/json` content-type * Customized integration responses can be specified for any content type in the `createIntegrationResponses` prop value.

Supplying any of these values without setting allowCreateOperation to true will result in an error. This pattern is the same for all four API operations.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

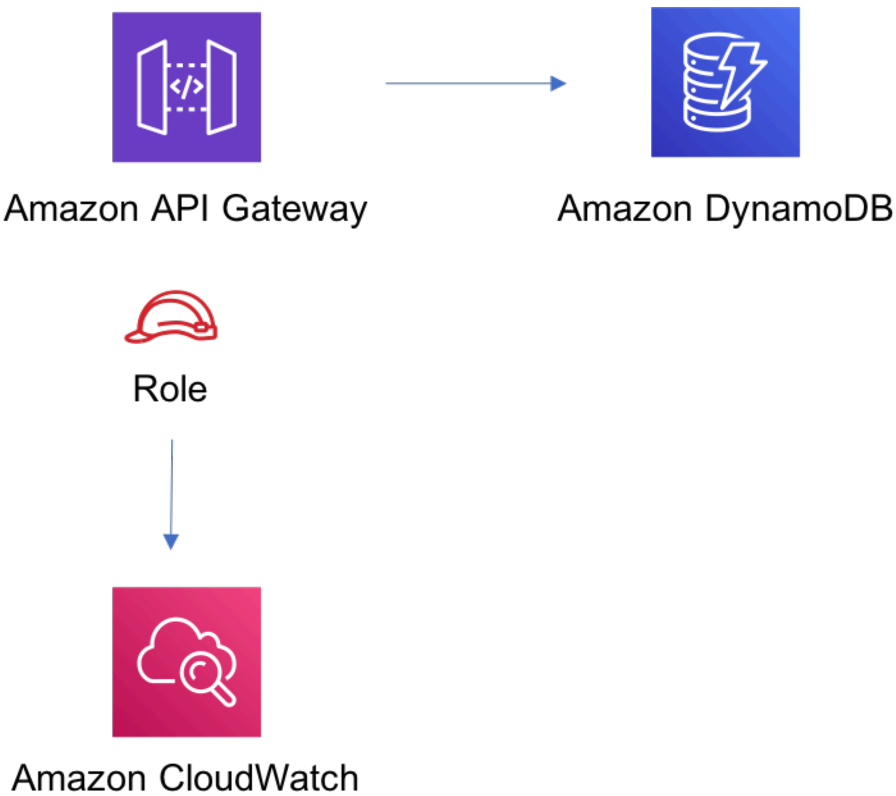
Amazon API Gateway

- Deploy an edge-optimized API endpoint
- Enable CloudWatch logging for API Gateway
- Configure least privilege access IAM role for API Gateway
- Set the default authorizationType for all API methods to IAM
- Enable X-Ray Tracing

Amazon DynamoDB Table

- Set the billing mode for DynamoDB Table to On-Demand (Pay per request)
- Enable server-side encryption for DynamoDB Table using AWS managed KMS Key
- Creates a partition key called "id" for DynamoDB Table
- Retain the Table when deleting the CloudFormation stack
- Enable continuous backups and point-in-time recovery

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-apigateway-dynamodb

aws-apigateway-iot

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_apigateway_iot
Typescript	@aws-solutions-constructs/aws-apigateway-iot
Java	software.amazon.awsconstructs.services.apigatewayiot

Overview

This AWS Solutions Construct implements an Amazon API Gateway REST API connected to AWS IoT pattern.

This construct creates a scalable HTTPS proxy between API Gateway and AWS IoT. This comes in handy when wanting to allow legacy devices that do not support the MQTT or MQTT/Websocket protocol to interact with the AWS IoT platform.

This implementation enables write-only messages to be published on given MQTT topics, and also supports shadow updates of HTTPS devices to allowed things in the device registry. It does not involve Lambda functions for proxying messages, and instead relies on direct API Gateway to AWS IoT integration which supports both JSON messages as well as binary messages.

Here is a minimal deployable pattern definition, note that the ATS endpoint for IoT must be used to avoid SSL certificate issues:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { ApiGatewayToIot } from '@aws-solutions-constructs/aws-apigateway-iot';

new ApiGatewayToIot(this, 'ApiGatewayToIotPattern', {
    iotEndpoint: 'a1234567890123-ats'
});
```

Python

```
from aws_solutions_constructs.aws_apigateway_iot import ApiGatewayToIot
from aws_cdk import Stack
from constructs import Construct

ApiGatewayToIot(self, 'ApiGatewayToIotPattern',
    iot_endpoint='a1234567890123-ats'
)
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.apigatewayiot.*;

new ApiGatewayToIot(this, "ApiGatewayToIotPattern", new
 ApiGatewayToIotProps.Builder()
        .iotEndpoint("a1234567890123-ats")
        .build());
```

Pattern Construct Props

Name	Type	Description
iotEndpoint	`string`	The AWS IoT endpoint subdomain to integrate the API Gateway with (e.g a1234567890123-ats). Note that this must point to the ATS endpoint to avoid SSL certificate trust issues. The endpoint can be retrieved by running `aws iot describe-endpoint --`

Name	Type	Description
		`endpoint-type iot:Data-ATS` .
apiGatewayCreateApiKey?	`boolean`	If set to `true`, an API Key is created and associated to a UsagePlan. User should specify `x-api-key` header while accessing RestApi. Default value set to `false`
apiGatewayExecutionRole?	[`iam.Role`]()	IAM Role used by the API Gateway to access AWS IoT. If not specified, a default role is created with wildcard ('*') access to all topics and things.
apiGatewayProps?	[`api.restApiProps`]()	Optional user-provided props to override the default props for the API Gateway.
logGroupProps?	[`logs.LogGroupProps`]()	User provided props to override the default props for for the CloudWatchLogs LogGroup.

Pattern Properties

Name	Type	Description
apiGateway	[`api.RestApi`]()	Returns an instance of the API Gateway REST API created by the pattern.
apiGatewayRole	[`iam.Role`]()	Returns an instance of the iam.Role created by the construct for API Gateway.

Name	Type	Description
apiGatewayCloudWatchRole?	iam.Role	Returns an instance of the iam.Role created by the construct for API Gateway for CloudWatch access.
apiGatewayLogGroup	logs.LogGroup	Returns an instance of the LogGroup created by the construct for API Gateway access logging to CloudWatch.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon API Gateway

- Deploy an edge-optimized API Endpoint
- Creates API Resources with `POST` Method to publish messages to IoT Topics
- Creates API Resources with `POST` Method to publish messages to ThingShadow & NamedShadows
- Enable CloudWatch logging for API Gateway
- Configure IAM role for API Gateway with access to all topics and things
- Set the default authorizationType for all API methods to IAM
- Enable X-Ray Tracing
- Creates a UsagePlan and associates to `prod` stage

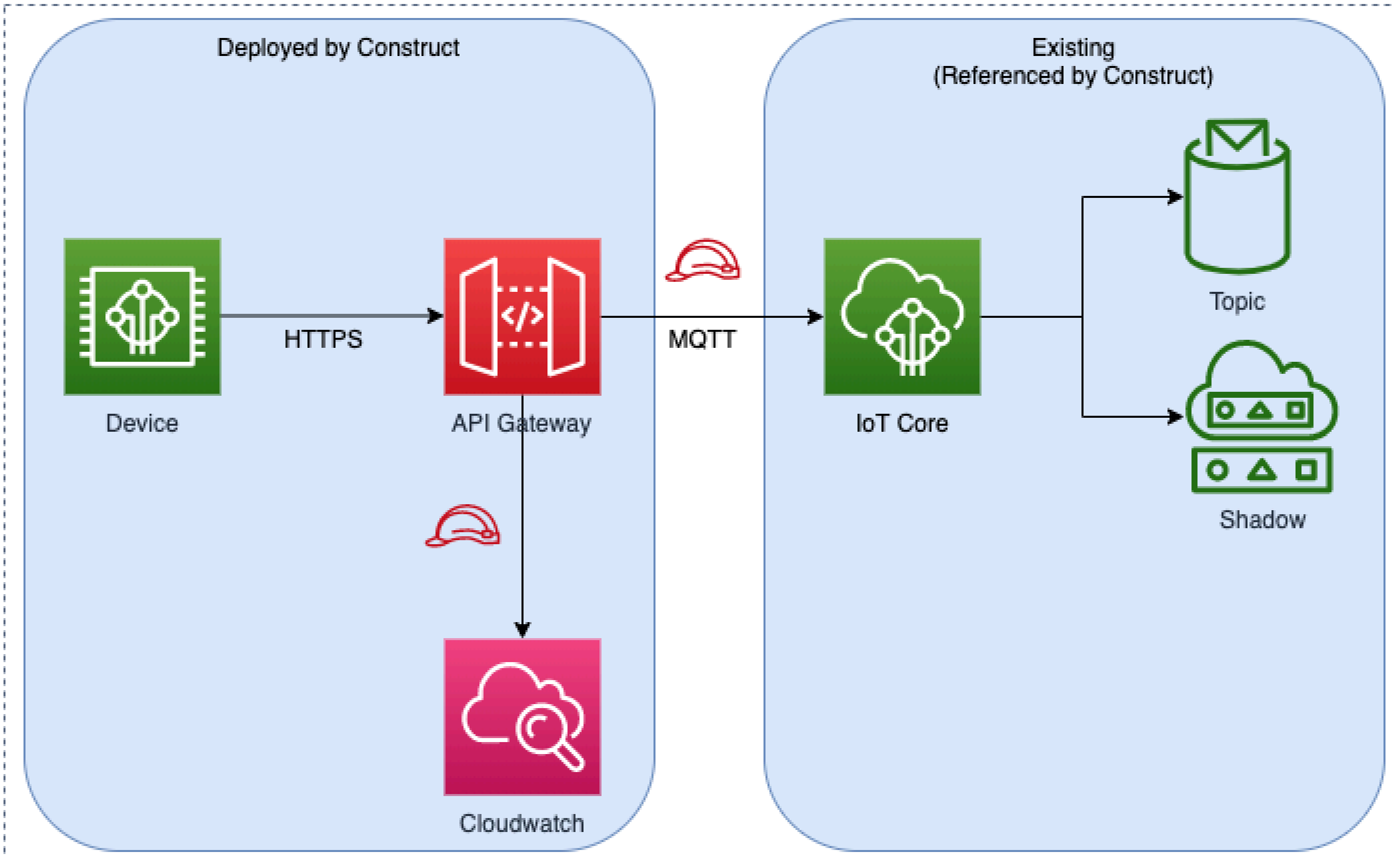
Below is a description of the different resources and methods exposed by the API Gateway after deploying the Construct.

Method	Resource	Query parameter(s)	Return code(s)	Description
POST	`/message/<topics>`	**qos**	`200/403/500`	By calling this endpoint, you

Method	Resource	Query parameter(s)	Return code(s)	Description
				need to pass the topics on which you would like to publish (e.g `/message/device/foo`).
POST	`/shadow/<thingName>`	**None**	`200/403/500`	This route allows to update the shadow document of a thing, given its `thingName` using Unnamed (classic) shadow type. The body shall comply with the standard shadow structure comprising a `state` node and associated `desired` and `reported` nodes.

Method	Resource	Query parameter(s)	Return code(s)	Description
POST	`/shadow/<thingName>/<shadowName>`	**None**	`200/403/500`	This route allows to update the named shadow document of a thing, given its `thingName` and the `shadowName` using the Named shadow type. The body shall comply with the standard shadow structure comprising a `state` node and associated `desired` and `reported` nodes.

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-apigateway-iot]()

aws-apigateway-kinesisstreams

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_apigateway_kinesisstreams`

Language	Package
Typescript	`@aws-solutions-constructs/aws-apigateway-kinesisstreams`
Java	`software.amazon.awsconstructs.services.apigatewaykinesisstreams`

Overview

This AWS Solutions Construct implements an Amazon API Gateway connected to an Amazon Kinesis Data Stream pattern.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { ApiGatewayToKinesisStreams, ApiGatewayToKinesisStreamsProps } from '@aws-solutions-constructs/aws-apigateway-kinesisstreams';

new ApiGatewayToKinesisStreams(this, 'test-apigw-kinesis', {});
```

Python

```
from aws_solutions_constructs.aws_apigateway_kinesisstreams import
 ApiGatewayToKinesisStreams
from aws_cdk import Stack
from constructs import Construct

ApiGatewayToKinesisStreams(self, 'test-apigw-kinesis')
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.apigatewaykinesisstreams.*;

new ApiGatewayToKinesisStreams(this, "test-apigw-kinesis", new
 ApiGatewayToKinesisStreamsProps.Builder()
        .build());
```

Pattern Construct Props

Name	Type	Description
apiGatewayProps?	`api.RestApiProps`	Optional user-provided props to override the default props for the API Gateway.
putRecordRequestTemplate?	`string`	API Gateway request template for the PutRecord action. If not provided, a default one will be used.
additionalPutRecordRequestTemplates?	`{ [contentType: string]: string; }`	Optional PutRecord Request Templates for content-types other than `application/json`. Use the `putRecordRequestTemplate` property to set the request template for the `application/json` content-type.
putRecordRequestModel?	`api.ModelOptions`	API Gateway request model for the PutRecord action. If not provided, a default one will be created.

Name	Type	Description
putRecordIntegrationResponses?	api.IntegrationResponses[]	Optional, custom API Gateway Integration Response for the PutRecord action.
putRecordsRequestTemplate?	string	API Gateway request template for the PutRecords action. If not provided, a default one will be used.
additionalPutRecordsRequestTemplates?	{ [contentType: string]: string; }	Optional PutRecords Request Templates for content-types other than `application/json`. Use the `putRecordsRequestTemplate` property to set the request template for the `application/json` content-type.
putRecordsRequestModel?	api.ModelOptions	
API Gateway request model for the PutRecords action. If not provided, a default one will be created.		
putRecordsIntegrationResponses?	api.IntegrationResponses[]	Optional, custom API Gateway Integration Response for the PutRecords action.
existingStreamObj?	kinesis.Stream	Existing instance of Kinesis Stream, providing both this and `kinesisStreamProps` will cause an error.

Name	Type	Description
kinesisStreamProps?	`kinesis.StreamProps`	Optional user-provided props to override the default props for the Kinesis stream.
logGroupProps?	`logs.LogGroupProps`	User provided props to override the default props for for the CloudWatchLogs LogGroup.
createCloudWatchAlarms	`boolean`	Whether to create recommend ed CloudWatch alarms for Kinesis Data Stream. Default value is set to `true`

Pattern Properties

Name	Type	Description
apiGateway	`api.RestApi`	Returns an instance of the API Gateway REST API created by the pattern.
apiGatewayRole	`iam.Role`	Returns an instance of the iam.Role created by the construct for API Gateway.
apiGatewayCloudWat chRole?	`iam.Role`	Returns an instance of the iam.Role created by the construct for API Gateway for CloudWatch access.
apiGatewayLogGroup	`logs.LogGroup`	Returns an instance of the LogGroup created by the construct for API Gateway access logging to CloudWatch.

Name	Type	Description
kinesisStream	kinesis.Stream	Returns an instance of the Kinesis stream created or used by the pattern.
cloudwatchAlarms?	[cloudwatch.Alarm[]](cloudwatch.Alarm[])	Returns an array of recommended CloudWatch Alarms created by the construct for Kinesis Data stream

Sample API Usage

Method	Request Path	Request Body	Stream Action	Description
POST	`/record`	`{ "data": "Hello World!", "partitionKey": "pk001" }`	`kinesis:PutRecord`	Writes a single data record into the stream.
POST	`/records`	`{ "records" : [{ "data": "abc", "partitionKey": "pk001" }, { "data": "xyz", "partitionKey": "pk001" }] }`	`kinesis:PutRecords`	Writes multiple data records into the stream in a single call.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

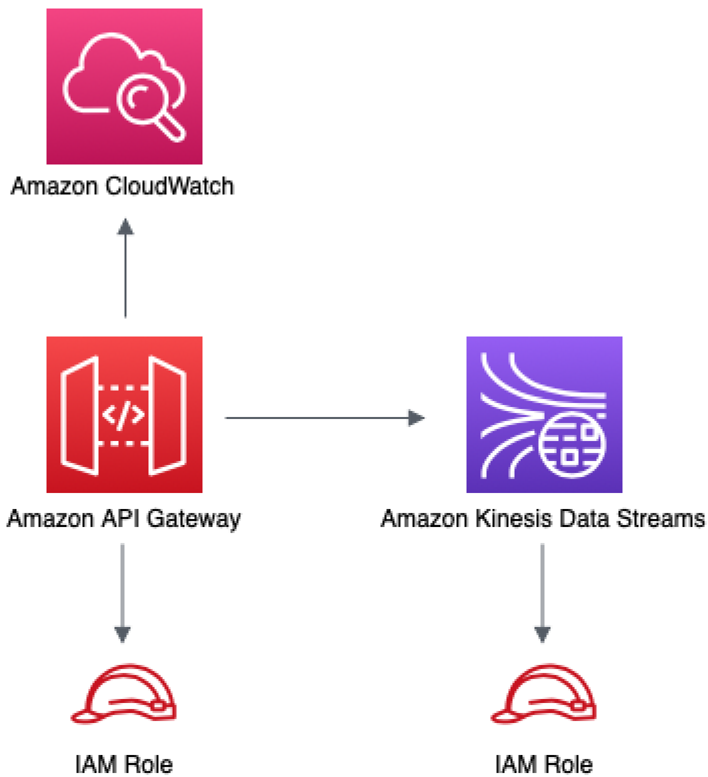
Amazon API Gateway

- Deploy an edge-optimized API endpoint
- Enable CloudWatch logging for API Gateway
- Configure least privilege access IAM role for API Gateway
- Set the default authorizationType for all API methods to IAM
- Enable X-Ray Tracing
- Validate request body before passing data to Kinesis

Amazon Kinesis Data Stream

- Configure least privilege access IAM role for Kinesis Stream
- Enable server-side encryption for Kinesis Stream using AWS Managed KMS Key

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-apigateway-kinesisstreams

aws-apigateway-lambda

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_apigateway_lambda`

Language	Package
Typescript	`@aws-solutions-constructs/aws-apigateway-lambda`
Java	`software.amazon.awsconstructs.services.apigatewaylambda`

Overview

This AWS Solutions Construct implements an Amazon API Gateway REST API connected to an AWS Lambda function pattern.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { ApiGatewayToLambda } from '@aws-solutions-constructs/aws-apigateway-lambda';
import * as lambda from 'aws-cdk-lib/aws-lambda';

new ApiGatewayToLambda(this, 'ApiGatewayToLambdaPattern', {
  lambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler',
    code: lambda.Code.fromAsset(`lambda`)
  }
});
```

Python

```
from aws_solutions_constructs.aws_apigateway_lambda import ApiGatewayToLambda
from aws_cdk import (
    aws_lambda as _lambda,
    Stack
)
```

```
from constructs import Construct

ApiGatewayToLambda(self, 'ApiGatewayToLambdaPattern',
            lambda_function_props=_lambda.FunctionProps(
                runtime=_lambda.Runtime.PYTHON_3_9,
                handler='index.handler',
                code=_lambda.Code.from_asset('lambda')
            )
            )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.apigatewaylambda.*;

new ApiGatewayToLambda(this, "ApiGatewayToLambdaPattern", new
 ApiGatewayToLambdaProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.

Name	Type	Description
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
apiGatewayProps?	`api.LambdaRestApiProps`	Optional user-provided props to override the default props for the API.
logGroupProps?	`logs.LogGroupProps`	User provided props to override the default props for for the CloudWatchLogs LogGroup.

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of the Lambda function created by the pattern.
apiGateway	`api.LambdaRestApi`	Returns an instance of the API Gateway REST API created by the pattern.
apiGatewayCloudWatchRole?	`iam.Role`	Returns an instance of the iam.Role created by the construct for API Gateway for CloudWatch access.
apiGatewayLogGroup	`logs.LogGroup`	Returns an instance of the LogGroup created by the construct for API Gateway access logging to CloudWatch.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

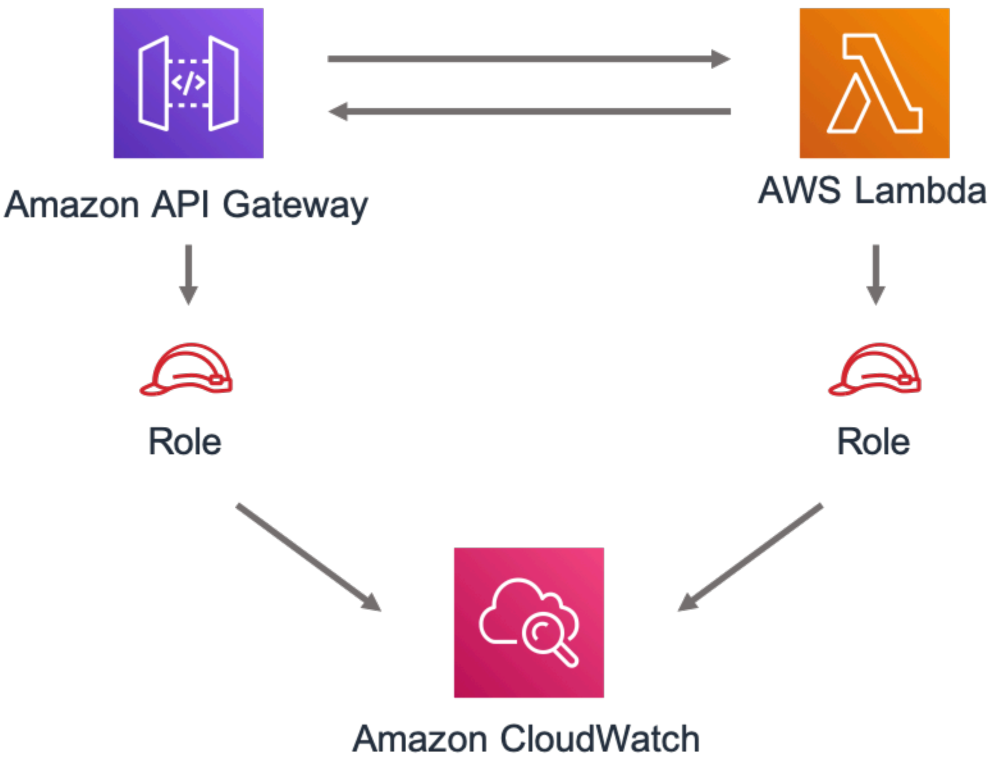
Amazon API Gateway

- Deploy an edge-optimized API endpoint
- Enable CloudWatch logging for API Gateway
- Configure least privilege access IAM role for API Gateway
- Set the default authorizationType for all API methods to IAM
- Enable X-Ray Tracing

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-apigateway-lambda

aws-apigateway-sagemakerendpoint

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_apigateway_sagemakerendpoint
Typescript	@aws-solutions-constructs/aws-apigateway-sagemakerendpoint
Java	software.amazon.awsconstructs.services.apigatewaysagemakerendpoint

Overview

This AWS Solutions Construct implements an Amazon API Gateway connected to an Amazon SageMaker endpoint pattern.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { ApiGatewayToSageMakerEndpoint, ApiGatewayToSageMakerEndpointProps } from
 '@aws-solutions-constructs/aws-apigateway-sagemakerendpoint';

// Below is an example VTL (Velocity Template Language) mapping template for mapping
 the Api GET request to the Sagemaker POST request
const requestTemplate = `
{
    "instances": [
        # set( $user_id = $input.params("user_id") )
        # set( $items = $input.params("items") )
        # foreach( $item in $items.split(",") )
        # if( $foreach.hasNext ),#end
        {"in0": [$user_id], "in1": [$item]}
            $esc.newline
        # end
    ]
```

```
}`

// Replace 'my-endpoint' with your Sagemaker Inference Endpoint
new ApiGatewayToSageMakerEndpoint(this, 'test-apigw-sagemakerendpoint', {
  endpointName: 'my-endpoint',
  resourcePath: '{user_id}',
  requestMappingTemplate: requestTemplate
});
```

Python

```
from aws_solutions_constructs.aws_apigateway_sagemakerendpoint import
 ApiGatewayToSageMakerEndpoint
from aws_cdk import Stack
from constructs import Construct

# Below is an example VTL (Velocity Template Language) mapping template for mapping
 the Api GET request to the Sagemaker POST request
request_template = """
{
    "instances": [
        # set( $user_id = $input.params("user_id") )
        # set( $items = $input.params("items") )
        # foreach( $item in $items.split(",") )
        # if( $foreach.hasNext ),#end
        {"in0": [$user_id], "in1": [$item]}
            $esc.newline
        # end
    ]
}"""

# Replace 'my-endpoint' with your Sagemaker Inference Endpoint
ApiGatewayToSageMakerEndpoint(self, 'test-apigw-sagemakerendpoint',
                                endpoint_name='my-endpoint',
                                resource_path='{user_id}',
                                request_mapping_template=request_template
                                )
```

Java

```
import software.constructs.Construct;
```

```
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.apigatewaysagemakerendpoint.*;

// Create an example VTL (Velocity Template Language) mapping template for mapping
 the Api GET request to the Sagemaker POST request
final String requestTemplate = "{"
        + "\"instances\": ["
        + "# set( $user_id = $input.params(\"user_id\") )"
        + "# set( $items = $input.params(\"items\") )"
        + "# foreach( $item in $items.split(\",\") )"
        + "# if( $foreach.hasNext ),#end"
        + "{\"in0\": [$user_id], \"in1\": [$item]}"
        + "    $esc.newline"
        + "# end"
        + "]"
        + "}";

// Replace ""my-endpoint"" with your Sagemaker Inference Endpoint
new ApiGatewayToSageMakerEndpoint(this, "ApiGatewayToSageMakerEndpointPattern",
        new ApiGatewayToSageMakerEndpointProps.Builder()
                .endpointName("my-endpoint")
                .resourcePath("{user_id}")
                .requestMappingTemplate(requestTemplate)
                .build());
```

Pattern Construct Props

Name	Type	Description
apiGatewayProps?	`api.RestApiProps`	Optional user-provided props to override the default props for the API Gateway.
apiGatewayExecutionRole?	`iam.Role`	IAM Role used by API Gateway to invoke the SageMaker endpoint. If not specified, a default role is created with access to `endpointName` .

Name	Type	Description
endpointName	`string`	Name of the deployed SageMaker inference endpoint.
resourceName?	`string`	Optional resource name where the GET method will be available.
resourcePath	`string`	Resource path for the GET method. The variable defined here can be referenced in `requestMappingTemplate` .
requestMappingTemplate	`string`	Mapping template to convert GET requests for the default `application/json` content-type received on the REST API to POST requests expected by the SageMaker endpoint.
additionalRequestTemplates	`{ [contentType: string]: string; }`	Optional Request Templates for content-types other than `application/json` . Use the `requestMappingTemplate` property to set the request template for the `application/json` content-type.
responseMappingTemplate?	`string`	Optional mapping template to convert responses received from the SageMaker endpoint.
logGroupProps?	[`logs.LogGroupProps`]()	User provided props to override the default props for for the CloudWatchLogs LogGroup.

Pattern Properties

Name	Type	Description
apiGateway	`api.RestApi`	Returns an instance of the API Gateway REST API created by the pattern.
apiGatewayRole	`iam.Role`	Returns an instance of the iam.Role created by the construct for API Gateway.
apiGatewayCloudWatchRole?	`iam.Role`	Returns an instance of the iam.Role created by the construct for API Gateway for CloudWatch access.
apiGatewayLogGroup	`logs.LogGroup`	Returns an instance of the LogGroup created by the construct for API Gateway access logging to CloudWatch.

Sample API Usage

Note: Each SageMaker endpoint is unique, and the response from the API will depend on the deployed model. The example given below assumes the sample from this blog post. For a reference on how that'd be implemented, please refer to integ.apigateway-sagemakerendpoint-overwrite.ts.

Method	Request Path	Query String	SageMaker Action	Description
GET	`/321`	`items=101,131,162`	`sagemaker:InvokeEndpoint`	Retrieves the predictions for a specific user and items.

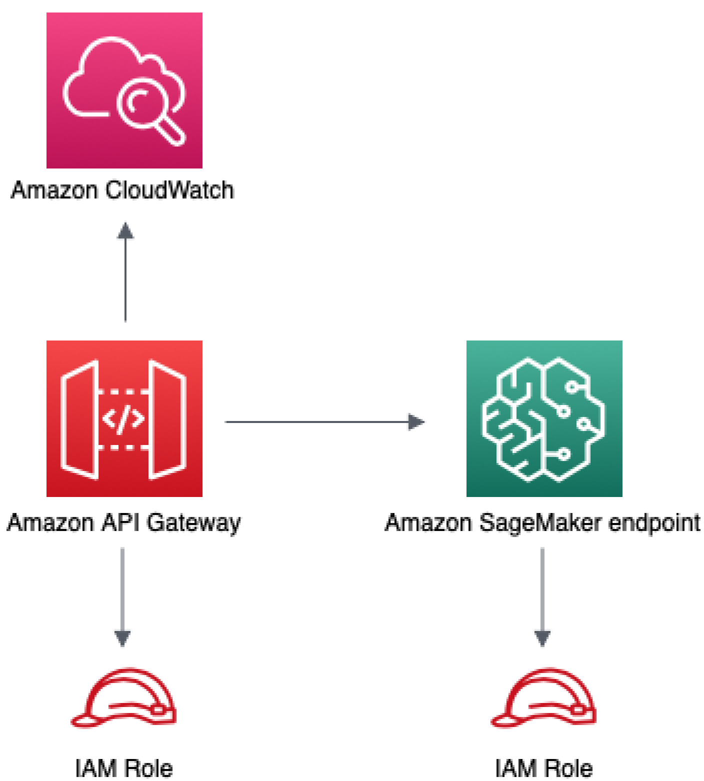
Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon API Gateway

- Deploy an edge-optimized API endpoint
- Enable CloudWatch logging for API Gateway
- Configure least privilege access IAM role for API Gateway
- Set the default authorizationType for all API methods to IAM
- Enable X-Ray Tracing
- Validate request parameters before passing data to SageMaker

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-apigateway-sagemakerendpoint

aws-apigateway-sqs

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_apigateway_sqs`
Typescript	`@aws-solutions-constructs/aws-apigateway-sqs`
Java	`software.amazon.awsconstructs.services.apigatewaysqs`

Overview

This AWS Solutions Construct implements an Amazon API Gateway connected to an Amazon SQS queue pattern.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { ApiGatewayToSqs, ApiGatewayToSqsProps } from "@aws-solutions-constructs/
aws-apigateway-sqs";
```

```
new ApiGatewayToSqs(this, 'ApiGatewayToSqsPattern', {});
```

Python

```
from aws_solutions_constructs.aws_apigateway_sqs import ApiGatewayToSqs
from aws_cdk import Stack
from constructs import Construct

ApiGatewayToSqs(self, 'ApiGatewayToSqsPattern')
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.apigatewaysqs.*;

new ApiGatewayToSqs(this, "ApiGatewayToSqsPattern", new
 ApiGatewayToSqsProps.Builder()
        .build());
```

Pattern Construct Props

Name	Type	Description
apiGatewayProps?	`api.RestApiProps`	Optional user-provided props to override the default props for the API Gateway.
queueProps?	`sqs.QueueProps`	Optional user-provided props to override the default props for the queue.
deployDeadLetterQueue?	`boolean`	Whether to deploy a secondary queue to be used as a dead letter queue. Defaults to `true`.

Name	Type	Description
maxReceiveCount	`number`	The number of times a message can be unsuccessfully dequeued before being moved to the dead-letter queue.
allowCreateOperation?	`boolean`	Whether to deploy an API Gateway Method for POST HTTP operations on the queue (i.e. sqs:SendMessage).
createRequestTemplate?	`string`	API Gateway Request Template for the create method for the default `application/json` content-type. This property is required if the `allowCreateOperation` property is set to true.
additionalCreateRequestTemplates?	`{ [contentType: string]: string; }`	Optional Create Request Templates for content-types other than `application/json`. Use the `createRequestTemplate` property to set the request template for the `application/json` content-type. This property can only be specified if the `allowCreateOperation` property is set to true.

Name	Type	Description
createIntegrationR esponses?	[api.IntegrationRes ponses[]](#)	Optional, custom API Gateway Integration Response for the create method. This property can only be specified if the `allowCreateOperation` property is set to true.
allowReadOperation?	`boolean`	Whether to deploy an API Gateway Method for GET HTTP operations on the queue (i.e. sqs:ReceiveMessage).
readRequestTemplate?	`string`	API Gateway Request Template for the read method for the default `application/json` content-type.
additionalReadRequ estTemplates?	`{ [contentType: string]: string; }`	Optional Read Request Templates for content-types other than `application/json`. Use the `readRequestTemplate` property to set the request template for the `application/json` content-type.
readIntegrationRes ponses?	[api.IntegrationRes ponses[]](#)	Optional, custom API Gateway Integration Response for the read method.
allowDeleteOperation?	`boolean`	Whether to deploy an API Gateway Method for HTTP DELETE operations on the queue (i.e. sqs:DeleteMessage).

Name	Type	Description
deleteRequestTemplate?	`string`	API Gateway Request Template for THE delete method for the default `application/json` content-type. This property can only be specified if the `allowDeleteOperation` property is set to true.
additionalDeleteRequestTemplates?	`{ [contentType: string]: string; }`	Optional Delete request templates for content-types other than `application/json`. Use the `deleteRequestTemplate` property to set the request template for the `application/json` content-type. This property can only be specified if the `allowDeleteOperation` property is set to true.
deleteIntegrationResponses?	[`api.IntegrationResponses[]`]()	Optional, custom API Gateway Integration Response for the delete method. This property can only be specified if the `allowDeleteOperation` property is set to true.
logGroupProps?	[`logs.LogGroupProps`]()	User provided props to override the default props for for the CloudWatchLogs LogGroup.

Name	Type	Description
enableEncryptionWithCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the queue is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined: queueProps.encryptionMasterKey, encryptionKey or encryptionKeyProps.
encryptionKey?	`kms.Key`	An optional, imported encryption key to encrypt the SQS Queue with.
encryptionKeyProps?	`kms.KeyProps`	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SQS queue with.

Pattern Properties

Name	Type	Description
apiGateway	`api.RestApi`	Returns an instance of the API Gateway REST API created by the pattern.
apiGatewayRole	`iam.Role`	Returns an instance of the iam.Role created by the construct for API Gateway.
apiGatewayCloudWatchRole?	`iam.Role`	Returns an instance of the iam.Role created by the

Name	Type	Description
		construct for API Gateway for CloudWatch access.
apiGatewayLogGroup	`logs.LogGroup`	Returns an instance of the LogGroup created by the construct for API Gateway access logging to CloudWatch.
sqsQueue	`sqs.Queue`	Returns an instance of the SQS queue created by the pattern.
deadLetterQueue?	`sqs.DeadLetterQueue`	Returns an instance of the DeadLetterQueue created by the pattern.

Sample API Usage

Method	Request Path	Request Body	Queue Action	Description
GET	`/`		`sqs::ReceiveMessage`	Retrieves a message from the queue.
POST	`/`	`{ "data": "Hello World!" }`	`sqs::SendMessage`	Delivers a message to the queue.
DELETE	`/message?receiptHandle=[value]`		`sqs::DeleteMessage`	Deletes a specified message from the queue

API Gateway Request/Response Template Properties Overview

This construct allows you to implement four DynamoDB API operations, CREATE/READ/DELETE (corresponding the HTTP POST/GET/DELETE requests respectively). They are completely independent and each follows the same pattern: * Setting `allowCreateOperation` to true will implement the `application/json` content-type with default request and response templates * The request template for `application/json` requests can be customized using the `createRequestTemplate` prop value * *Additional* request templates can be specified using the `additionalCreateRequestTemplates` prop value. Note - these DO NOT replace the `application/json` content-type * Customized integration responses can be specified for any content type in the `createIntegrationResponses` prop value.

Supplying any of these values without setting allowCreateOperation to true will result in an error. This pattern is the same for all four API operations.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

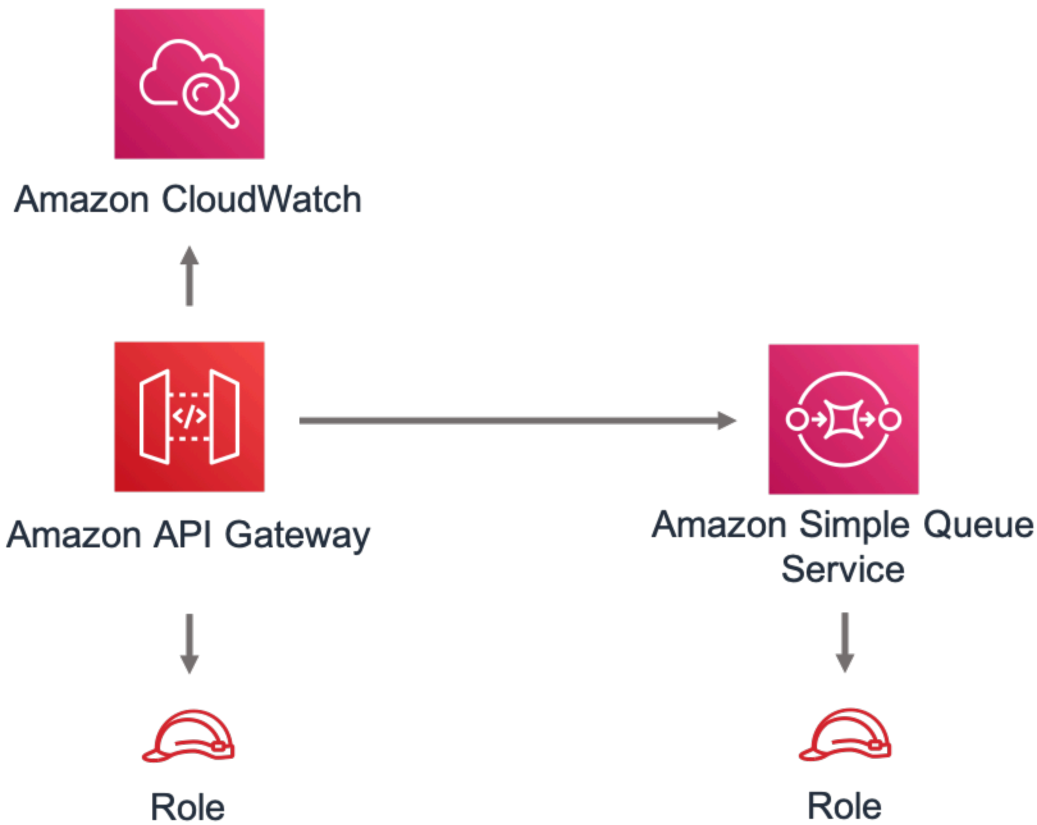
Amazon API Gateway

- Deploy an edge-optimized API endpoint
- Enable CloudWatch logging for API Gateway
- Configure least privilege access IAM role for API Gateway
- Set the default authorizationType for all API methods to IAM
- Enable X-Ray Tracing

Amazon SQS Queue

- Deploy SQS dead-letter queue for the source SQS Queue
- Enable server-side encryption for source SQS Queue using AWS Managed KMS Key
- Enforce encryption of data in transit

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-apigateway-sqs](https://github.com/awslabs/aws-solutions-constructs/tree/main/source/patterns/%40aws-solutions-constructs/aws-apigateway-sqs)

aws-cloudfront-apigateway-lambda

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_cloudfront_apigateway_lambda`

Language	Package
Typescript	@aws-solutions-constructs/aws-cloudfront-apigateway-lambda
Java	software.amazon.awsconstructs.services.cloudfrontapigatewaylambda

Overview

This AWS Solutions Construct implements an AWS CloudFront fronting an Amazon API Gateway Lambda backed REST API.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { CloudFrontToApiGatewayToLambda } from '@aws-solutions-constructs/aws-cloudfront-apigateway-lambda';
import * as lambda from 'aws-cdk-lib/aws-lambda';

new CloudFrontToApiGatewayToLambda(this, 'test-cloudfront-apigateway-lambda', {
  lambdaFunctionProps: {
    code: lambda.Code.fromAsset(`lambda`),
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler'
  },
  apiGatewayProps: {
    defaultMethodOptions: {
      authorizationType: api.AuthorizationType.NONE
    }
  },
});
```

Python

```
from aws_solutions_constructs.aws_cloudfront_apigateway_lambda import
 CloudFrontToApiGatewayToLambda
from aws_cdk import (
  aws_lambda as _lambda,
  aws_apigateway as apigw,
  Stack
)
from constructs import Construct

        CloudFrontToApiGatewayToLambda(
            self, 'CloudFrontApiGatewayToLambda',
            lambda_function_props=_lambda.FunctionProps(
                runtime=_lambda.Runtime.PYTHON_3_7,
                code=_lambda.Code.from_asset('lambda'),
                handler='hello.handler',
            ),
            # NOTE - we use RestApiProps here because the actual type,
 LambdaRestApiProps requires
            # the handler function which does not yet exist. As RestApiProps is a
 subset of of LambdaRestApiProps
            # (although does not *extend* that interface) this works fine when the
 props object reaches the
            # underlying TypeScript code that implements Constructs
            api_gateway_props=apigw.RestApiProps(
                default_method_options=apigw.MethodOptions(
                    authorization_type=apigw.AuthorizationType.NONE
                )
            )
        )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.cloudfrontapigatewaylambda.*;
import
 software.amazon.awsconstructs.services.cloudfrontapigatewaylambda.CloudFrontToApiGatewayToL
```

```
new CloudFrontToApiGatewayToLambda(this, "ApiGatewayToLambdaPattern", new
 CloudFrontToApiGatewayToLambdaProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X) // execution environment
                .code(Code.fromAsset("lambda")) // code loaded from the `lambda`
 directory (under root, next to `src`)
                .handler("hello.handler") // file is `hello`, function is `handler`
                .build())
        // NOTE - we use RestApiProps here because the actual type,
 LambdaRestApiProps requires
        // the handler function which does not yet exist. As RestApiProps is a
 subset of of LambdaRestApiProps
        // (although does not *extend* that interface) this works fine when the
 props object reaches the
        // underlying TypeScript code that implements Constructs
        .apiGatewayProps(new RestApiProps.Builder()
                .defaultMethodOptions(new MethodOptions.Builder()
                        .authorizationType(AuthorizationType.NONE)
                        .build())
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	Optional user provided props to override the default props for the Lambda function.
apiGatewayProps?	`api.LambdaRestApiProps`	User provided props to override the default props for the API Gateway. As of release 2.48.0, clients must

Name	Type	Description
		include this property with `defaultMethodOptions: { authorizationType: string }` specified. See Issue1043 in the github repo https://github.com/awslabs/aws-solutions-constructs/issues/1043
cloudFrontDistributionProps?	[`cloudfront.DistributionProps`]()	Optional user provided props to override the default props for CloudFront Distribution
insertHttpSecurityHeaders?	`boolean`	Optional user provided props to turn on/off the automatic injection of best practice HTTP security headers in all responses from CloudFront
responseHeadersPolicyProps?	[`cloudfront.ResponseHeadersPolicyProps`]()	Optional user provided configuration that cloudfront applies to all http responses.
logGroupProps?	[`logs.LogGroupProps`]()	Optional user provided props to override the default props for for the CloudWatchLogs LogGroup.
cloudFrontLoggingBucketProps?	[`s3.BucketProps`]()	Optional user provided props to override the default props for the CloudFront Logging Bucket.

Pattern Properties

Name	Type	Description
cloudFrontWebDistribution	cloudfront.Distribution	Returns an instance of cloudfront.Distribution created by the construct
cloudFrontFunction?	cloudfront.Function	Returns an instance of the Cloudfront function created by the pattern.
cloudFrontLoggingBucket	s3.Bucket	Returns an instance of the logging bucket for CloudFront Distribution.
apiGateway	api.RestApi	Returns an instance of the API Gateway REST API created by the pattern.
apiGatewayCloudWatchRole?	iam.Role	Returns an instance of the iam.Role created by the construct for API Gateway for CloudWatch access.
apiGatewayLogGroup	logs.LogGroup	Returns an instance of the LogGroup created by the construct for API Gateway access logging to CloudWatch.
lambdaFunction	lambda.Function	Returns an instance of the Lambda function created by the pattern.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon CloudFront

- Configure Access logging for CloudFront Distribution
- Enable automatic injection of best practice HTTP security headers in all responses from CloudFront Distribution

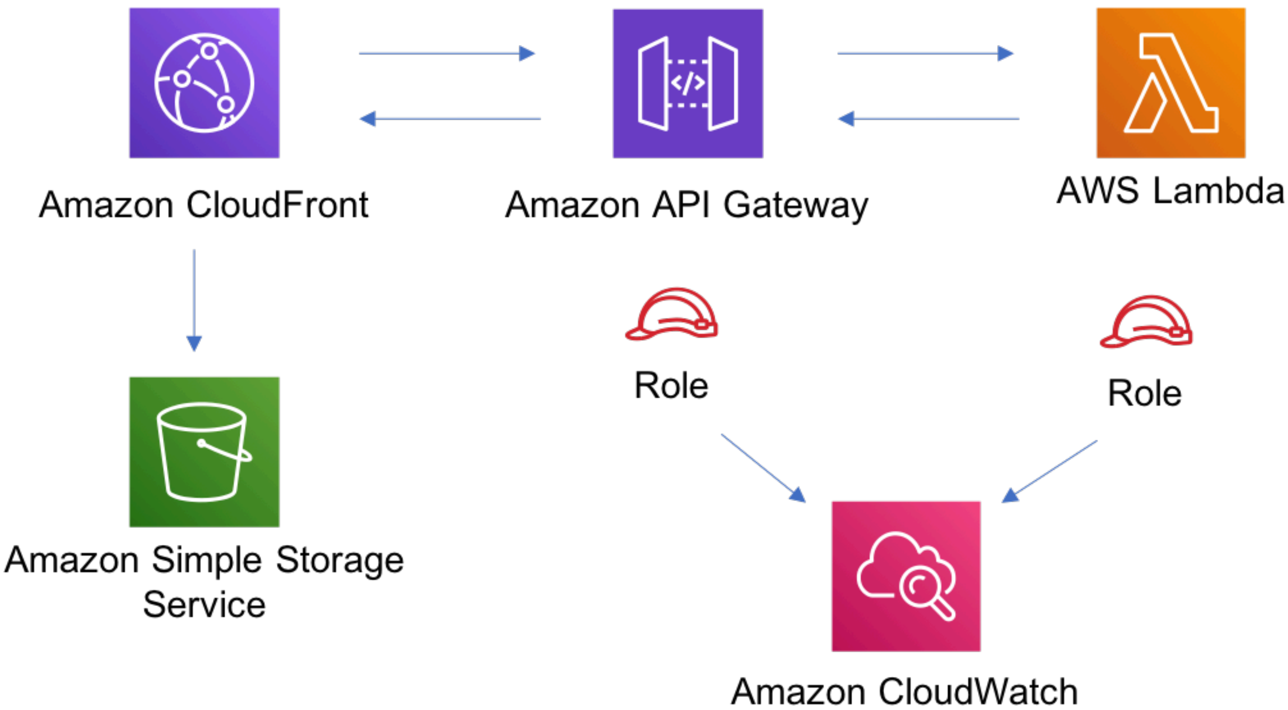
Amazon API Gateway

- Deploy a regional API endpoint
- Enable CloudWatch logging for API Gateway
- Configure least privilege access IAM role for API Gateway
- Set the default authorizationType for all API methods to NONE
- Enable X-Ray Tracing

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-cloudfront-apigateway-lambda

aws-cloudfront-apigateway

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_cloudfront_apigateway`
Typescript	`@aws-solutions-constructs/aws-cloudfront-apigateway`

Language	Package
Java	`software.amazon.awsconstructs.services.cloudfrontapigateway`

Overview

This AWS Solutions Construct implements an AWS CloudFront fronting an Amazon API Gateway REST API.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { CloudFrontToApiGateway } from '@aws-solutions-constructs/aws-cloudfront-apigateway';
import * as lambda from 'aws-cdk-lib/aws-lambda';
import * as api from 'aws-cdk-lib/aws-apigateway';

const lambdaProps: lambda.FunctionProps = {
  code: lambda.Code.fromAsset(`lambda`),
  runtime: lambda.Runtime.NODEJS_16_X,
  handler: 'index.handler'
};

const lambdafunction = new lambda.Function(this, 'LambdaFunction', lambdaProps);

const apiGatewayProps: api.LambdaRestApiProps = {
  handler: lambdafunction,
  endpointConfiguration: {
    types: [api.EndpointType.REGIONAL]
  },
  defaultMethodOptions: {
    authorizationType: api.AuthorizationType.NONE
  }
};

const apiGateway = new api.LambdaRestApi(this, 'LambdaRestApi', apiGatewayProps);
```

```
new CloudFrontToApiGateway(this, 'test-cloudfront-apigateway', {
  existingApiGatewayObj: apiGateway
});
```

Python

```
from aws_solutions_constructs.aws_cloudfront_apigateway import
 CloudFrontToApiGateway
from aws_cdk import (
    aws_lambda as _lambda,
    aws_apigateway as api,
    Stack
)
from constructs import Construct

lambda_function = _lambda.Function(self, 'LambdaFunction',
                                    code=_lambda.Code.from_asset(
                                        'lambda'),
                                    runtime=_lambda.Runtime.PYTHON_3_9,
                                    handler='index.handler')

api_gateway = api.LambdaRestApi(self, 'LambdaRestApi',
                                handler=lambda_function,
                                endpoint_configuration=api.EndpointConfiguration(
                                    types=[api.EndpointType.REGIONAL]
                                ),
                                default_method_options=api.MethodOptions(
                                    authorization_type=api.AuthorizationType.NONE
                                ))

CloudFrontToApiGateway(self, 'test-cloudfront-apigateway',
                        existing_api_gateway_obj=api_gateway
                        )
```

Java

```
import software.constructs.Construct;
import java.util.List;

import software.amazon.awscdk.Stack;
```

```
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awscdk.services.apigateway.*;
import software.amazon.awsconstructs.services.cloudfrontapigateway.*;

final Function lambdaFunction = Function.Builder.create(this, "IndexHandler")
        .runtime(Runtime.NODEJS_16_X)
        .code(Code.fromAsset("lambda"))
        .handler("index.handler")
        .build();

final LambdaRestApi apiGateway = LambdaRestApi.Builder.create(this, "myapi")
        .handler(lambdaFunction)
        .endpointConfiguration(new EndpointConfiguration.Builder()
                .types(List.of(EndpointType.REGIONAL))
                .build())
        .build();

new CloudFrontToApiGateway(this, "test-cloudfront-apigateway", new
 CloudFrontToApiGatewayProps.Builder()
        .existingApiGatewayObj(apiGateway)
        .build());
```

Pattern Construct Props

Name	Type	Description
existingApiGatewayObj	api.RestApi	The regional API Gateway that will be fronted with the CloudFront
cloudFrontDistributionProps?	cloudfront.DistributionProps | any	Optional user provided props to override the default props for CloudFront Distribution
insertHttpSecurityHeaders?	boolean	Optional user provided props to turn on/off the automatic injection of best practice HTTP

Name	Type	Description
		security headers in all responses from CloudFront
responseHeadersPolicyProps?	cloudfront.ResponseHeadersPolicyProps	Optional user provided configuration that cloudfront applies to all http responses.
cloudFrontLoggingBucketProps?	s3.BucketProps	Optional user provided props to override the default props for the CloudFront Logging Bucket.

Pattern Properties

Name	Type	Description
cloudFrontWebDistribution	cloudfront.Distribution	Returns an instance of cloudfront.Distribution created by the construct
apiGateway	api.RestApi	Returns an instance of the API Gateway REST API created by the pattern.
cloudFrontFunction?	cloudfront.Function	Returns an instance of the Cloudfront function created by the pattern.
cloudFrontLoggingBucket	s3.Bucket	Returns an instance of the logging bucket for CloudFront Distribution.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

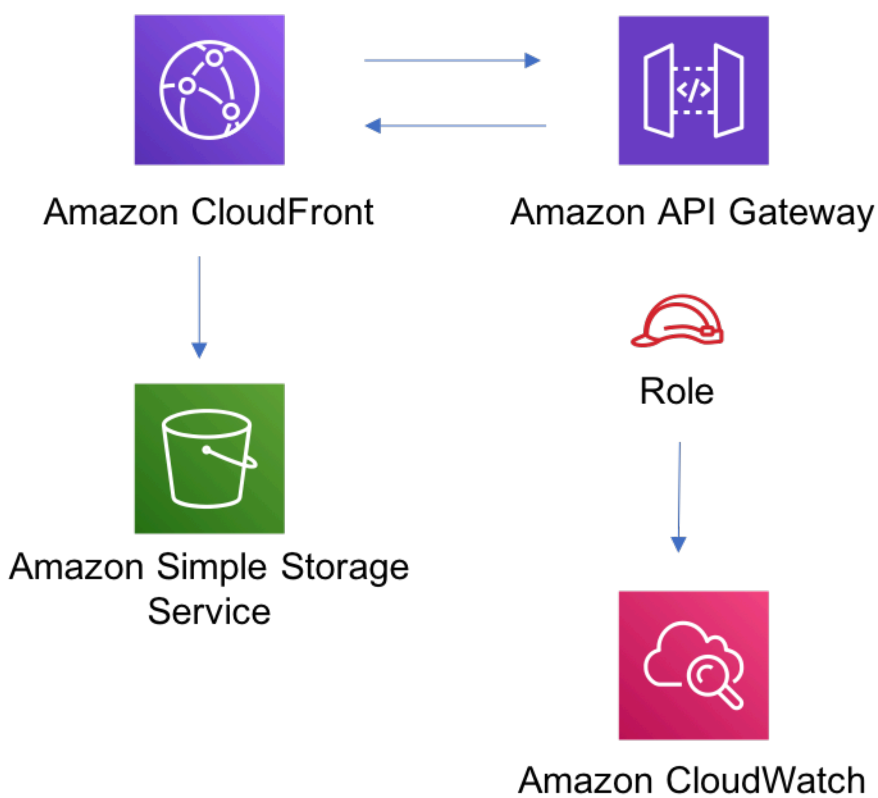
Amazon CloudFront

- Configure Access logging for CloudFront Distribution
- Enable automatic injection of best practice HTTP security headers in all responses from CloudFront Distribution

Amazon API Gateway

- User provided API Gateway object is used as-is
- Enable X-Ray Tracing

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-cloudfront-apigateway

aws-cloudfront-mediastore

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_cloudfront_mediastore`
TypeScript	`@aws-solutions-constructs/aws-cloudfront-mediastore`
Java	`software.amazon.awsconstructs.services.cloudfrontmediastore`

Overview

This AWS Solutions Construct implements an Amazon CloudFront distribution to an AWS Elemental MediaStore container.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { CloudFrontToMediaStore } from '@aws-solutions-constructs/aws-cloudfront-mediastore';
```

```
new CloudFrontToMediaStore(this, 'test-cloudfront-mediastore-default', {});
```

Python

```
from aws_solutions_constructs.aws_cloudfront_mediastore import
 CloudFrontToMediaStore
from aws_cdk import Stack
from constructs import Construct

CloudFrontToMediaStore(self, 'test-cloudfront-mediastore-default')
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.cloudfrontmediastore.*;

new CloudFrontToMediaStore(this, "test-cloudfront-mediastore-default", new
 CloudFrontToMediaStoreProps.Builder()
        .build());
```

Pattern Construct Props

Name	Type	Description
existingMediaStore ContainerObj?	`mediastore.CfnContainer`	Optional user provided MediaStore container to override the default MediaStore container.
mediaStoreContainerProps?	`mediastore.CfnContainerProps`	Optional user provided props to override the default props for the MediaStore Container.

Name	Type	Description
cloudFrontDistributionProps?	`cloudfront.DistributionProps` |any	Optional user provided props to override the default props for the CloudFront Distribution.
insertHttpSecurityHeaders?	`boolean`	Optional user provided props to turn on/off the automatic injection of best practice HTTP security headers in all responses from CloudFront
responseHeadersPolicyProps?	`cloudfront.ResponseHeadersPolicyProps`	Optional user provided configuration that cloudfront applies to all http responses.
cloudFrontLoggingBucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the CloudFront Logging Bucket.

Pattern Properties

Name	Type	Description
cloudFrontWebDistribution	`cloudfront.Distribution`	Returns an instance of cloudfront.Distribution created by the construct.
mediaStoreContainer	`mediastore.CfnContainer`	Returns an instance of mediastore.CfnContainer.
cloudFrontLoggingBucket	`s3.Bucket`	Returns an instance of s3.Bucket as the logging bucket for the CloudFront Web Distribution.

Name	Type	Description
cloudFrontOriginRequestPolicy	`cloudfront.OriginRequestPolicy`	Returns an instance of cloudfront.OriginRequestPolicy created by the construct for the CloudFront Web Distribution.
cloudFrontOriginAccessIdentity?	`cloudfront.OriginAccessIdentity`	Returns an instance of cloudfront.OriginAccessIdentity created by the construct for the CloudFront Web Distribution origin custom headers and the MediaStore Container policy.
cloudFrontFunction?	`cloudfront.Function`	Returns an instance of the Cloudfront function created by the pattern.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon CloudFront

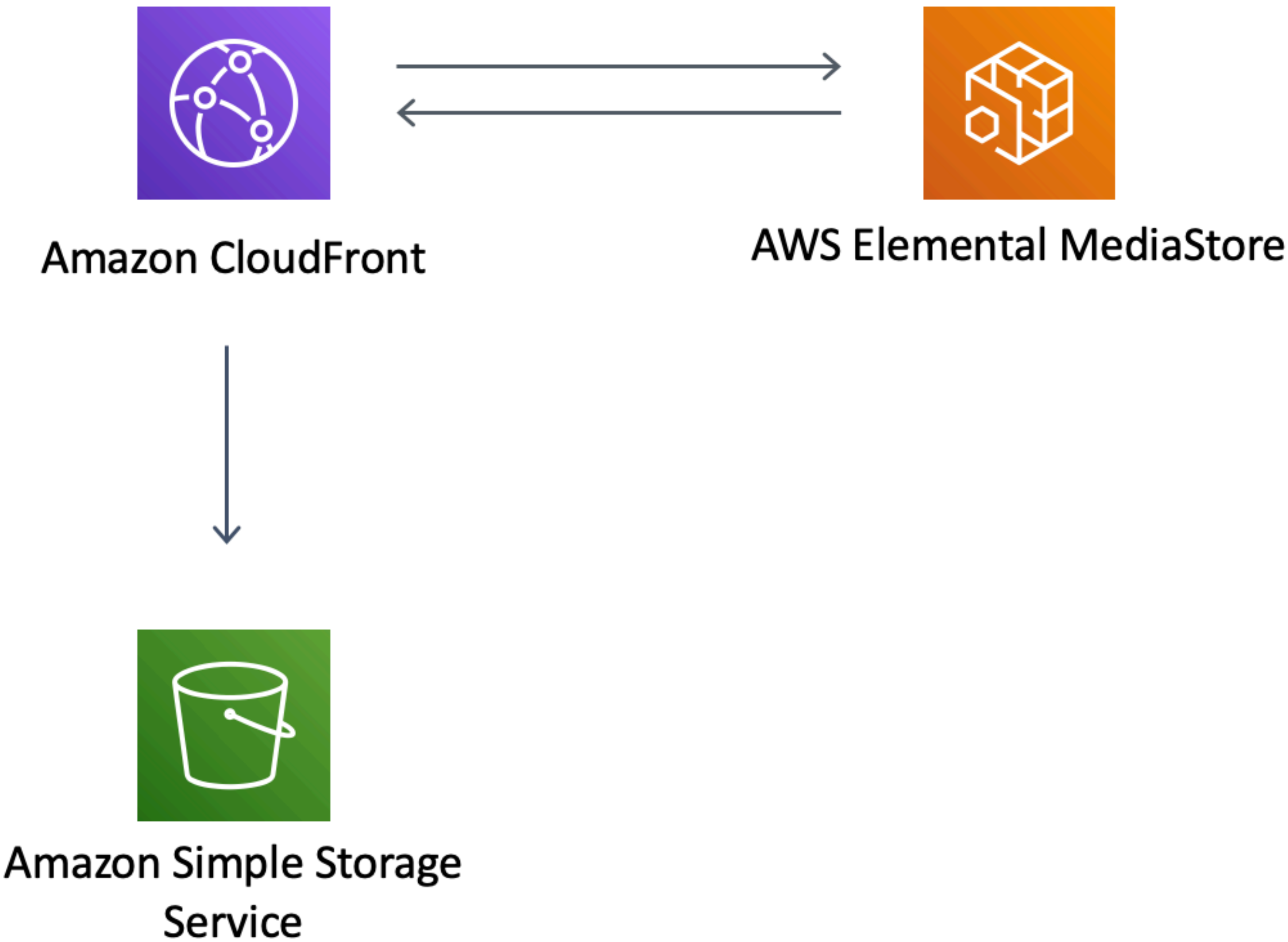
- Configure access logging for CloudFront Web Distribution
- Enable CloudFront Origin Request Policy for AWS Elemental MediaStore Container
- Set `User-Agent` custom header with CloudFront Origin Access Identity
- Enable automatic injection of best practice HTTP security headers in all responses from CloudFront Distribution

AWS Elemental MediaStore

- Set the deletion policy to retain the resource
- Set the container name with the CloudFormation stack name
- Set the default Container Cross-origin resource sharing (CORS) policy
- Set the default Object Life Cycle policy

- Set the default Container Policy to allow only `aws:UserAgent` with CloudFront Origin Access Identity
- Set the default Metric Policy
- Enable the access logging

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-cloudfront-mediastore

aws-cloudfront-s3

CFN-RESOURCES STABLE

Language	Package
Python	aws_solutions_constructs.aws_cloudfront_s3
Typescript	@aws-solutions-constructs/aws-cloudfront-s3
Java	software.amazon.awsconstructs.services.cloudfronts3

Overview

This AWS Solutions Construct provisions an Amazon CloudFront Distribution that serves objects from an AWS S3 Bucket via an Origin Access Control (OAC).

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { CloudFrontToS3 } from '@aws-solutions-constructs/aws-cloudfront-s3';

new CloudFrontToS3(this, 'test-cloudfront-s3', {});
```

Python

```
from aws_solutions_constructs.aws_cloudfront_s3 import CloudFrontToS3
from aws_cdk import Stack
from constructs import Construct

CloudFrontToS3(self, 'test-cloudfront-s3')
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.cloudfronts3.*;

new CloudFrontToS3(this, "test-cloudfront-s3", new CloudFrontToS3Props.Builder()
        .build());
```

Pattern Construct Props

Name	Type	Description
existingBucketObj?	`s3.IBucket`	Existing instance of S3 Bucket object or interface. If this is provided, then also providing bucketProps will cause an error.
bucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the S3 Bucket.
cloudFrontDistributionProps?	`cloudfront.DistributionProps`	Optional user provided props to override the default props for CloudFront Distribution
insertHttpSecurityHeaders?	`boolean`	Optional user provided props to turn on/off the automatic injection of best practice HTTP security headers in all responses from CloudFront
responseHeadersPolicyProps?	`cloudfront.ResponseHeadersPolicyProps`	Optional user provided configuration that cloudfront applies to all http responses.

Name	Type	Description
originPath?	`string`	Optional user provided props to provide an[originPath](#) that CloudFront appends to the origin domain name when CloudFront requests content from the origin. The string should start with a /, for example: `/production` . Default value is `'/'`
loggingBucketProps?	[`s3.BucketProps`](#)	Optional user provided props to override the default props for the S3 Logging Bucket.
cloudFrontLoggingBucketProps?	[`s3.BucketProps`](#)	Optional user provided props to override the default props for the CloudFront Logging Bucket.
logS3AccessLogs?	boolean	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with associated storage costs for the logs. Enabling Access Logging is a best practice. default - true

Pattern Properties

Name	Type	Description
cloudFrontWebDistribution	[`cloudfront.Distribution`](#)	Returns an instance of cloudfront.Distribution created by the construct.

Name	Type	Description
cloudFrontFunction?	`cloudfront.Function`	Returns an instance of the Cloudfront function created by the construct.
cloudFrontLoggingBucket	`s3.Bucket`	Returns an instance of the logging bucket for the CloudFront Distribution.
s3BucketInterface	`s3.IBucket`	Returns an instance of s3.IBucke t created by the construct.
s3Bucket?	`s3.Bucket`	Returns an instance of s3.Bucket created by the construct. IMPORTANT: If `existingBucketObj` was provided in Pattern Construct Props, this property will be `undefined`
s3LoggingBucket?	`s3.Bucket`	Returns an instance of s3.Bucket created by the construct as the logging bucket for the primary bucket.
originAccessControl?	`cloudfront.CfnOriginAccessControl`	Returns an instance of cloudfront.CfnOriginAccessC ontrol created by the construct.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon CloudFront

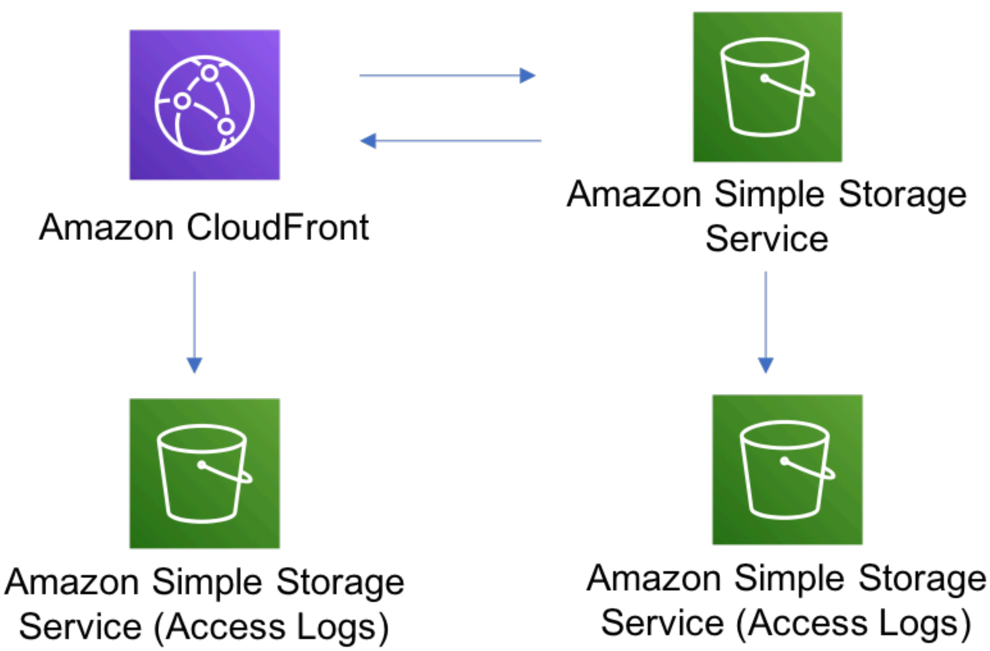
- Configure Access logging for CloudFront Distribution

- Enable automatic injection of best practice HTTP security headers in all responses from CloudFront Distribution
- CloudFront originPath set to `'/'`

Amazon S3 Bucket

- Configure Access logging for S3 Bucket
- Enable server-side encryption for S3 Bucket using AWS managed KMS Key
- Enforce encryption of data in transit
- Turn on the versioning for S3 Bucket
- Don't allow public access for S3 Bucket
- Retain the S3 Bucket when deleting the CloudFormation stack
- Applies Lifecycle rule to move noncurrent object versions to Glacier storage after 90 days

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-cloudfront-s3](#)

aws-cognito-apigateway-lambda

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_cognito_apigateway_lambda`
Typescript	`@aws-solutions-constructs/aws-cognito-apigateway-lambda`
Java	`software.amazon.awsconstructs.services.cognitoapigatewaylambda`

Overview

This AWS Solutions Construct implements an Amazon Cognito securing an Amazon API Gateway Lambda backed REST APIs pattern.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { CognitoToApiGatewayToLambda } from '@aws-solutions-constructs/aws-cognito-
apigateway-lambda';
import * as lambda from 'aws-cdk-lib/aws-lambda';

new CognitoToApiGatewayToLambda(this, 'test-cognito-apigateway-lambda', {
  lambdaFunctionProps: {
    code: lambda.Code.fromAsset(`lambda`),
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler'
  }
});
```

Python

```
from aws_solutions_constructs.aws_cognito_apigateway_lambda import
 CognitoToApiGatewayToLambda
from aws_cdk import (
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

CognitoToApiGatewayToLambda(self, 'test-cognito-apigateway-lambda',
                            lambda_function_props=_lambda.FunctionProps(
                                code=_lambda.Code.from_asset('lambda'),
                                runtime=_lambda.Runtime.PYTHON_3_9,
                                handler='index.handler'
                            )
                            )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
```

```
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.cognitoapigatewaylambda.*;

new CognitoToApiGatewayToLambda(this, "test-cognito-apigateway-lambda",
        new CognitoToApiGatewayToLambdaProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_16_X)
                        .code(Code.fromAsset("lambda"))
                        .handler("index.handler")
                        .build())
                .build());
```

If you are defining resources and methods on your API (e.g. proxy = false), then you must call addAuthorizers() after the API is fully defined to ensure every method is protected. Here is an example:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { CognitoToApiGatewayToLambda } from '@aws-solutions-constructs/aws-cognito-
apigateway-lambda';
import * as lambda from 'aws-cdk-lib/aws-lambda';

const construct = new CognitoToApiGatewayToLambda(this, 'test-cognito-apigateway-
lambda', {
    lambdaFunctionProps: {
        code: lambda.Code.fromAsset(`lambda`),
        runtime: lambda.Runtime.NODEJS_16_X,
        handler: 'index.handler'
    },
    apiGatewayProps: {
        proxy: false
    }
});

const resource = construct.apiGateway.root.addResource('foobar');
resource.addMethod('POST');

// Mandatory to call this method to Apply the Cognito Authorizers on all API methods
```

```
construct.addAuthorizers();
```

Python

```
from aws_solutions_constructs.aws_cognito_apigateway_lambda import
 CognitoToApiGatewayToLambda
from aws_cdk import (
    aws_lambda as _lambda,
    aws_apigateway as api,
    Stack
)
from constructs import Construct
from typing import Any

# Overriding LambdaRestApiProps with type Any
gateway_props = dict[Any, Any]

construct = CognitoToApiGatewayToLambda(self, 'test-cognito-apigateway-lambda',
                                        lambda_function_props=_lambda.FunctionProps(
                                            code=_lambda.Code.from_asset(
                                                'lambda'),
                                            runtime=_lambda.Runtime.PYTHON_3_9,
                                            handler='index.handler'
                                        ),
                                        api_gateway_props=gateway_props(
                                            proxy=False
                                        )
                                        )

resource = construct.api_gateway.root.add_resource('foobar')
resource.add_method('POST')

# Mandatory to call this method to Apply the Cognito Authorizers on all API methods
construct.add_authorizers()
```

Java

```
import software.constructs.Construct;

import java.util.HashMap;
import java.util.Map;
```

```
import java.util.Optional;

import software.amazon.awscdk.*;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awscdk.services.apigateway.IResource;
import software.amazon.awsconstructs.services.cognitoapigatewaylambda.*;

// Overriding LambdaRestApiProps with type Any
Map<String, Optional<?>> gatewayProps = new HashMap<String, Optional<?>>();
gatewayProps.put("proxy", Optional.of(false));

final CognitoToApiGatewayToLambda construct = new CognitoToApiGatewayToLambda(this,
        "test-cognito-apigateway-lambda",
        new CognitoToApiGatewayToLambdaProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_16_X)
                        .code(Code.fromAsset("lambda"))
                        .handler("index.handler")
                        .build())
                .apiGatewayProps(gatewayProps)
                .build());

final IResource resource =
 construct.getApiGateway().getRoot().addResource("foobar");
resource.addMethod("POST");

// Mandatory to call this method to Apply the Cognito Authorizers on all API methods
construct.addAuthorizers();
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.

Name	Type	Description
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
apiGatewayProps?	`api.LambdaRestApiProps`	Optional user provided props to override the default props for API Gateway
cognitoUserPoolProps?	`cognito.UserPoolProps`	Optional user provided props to override the default props for Cognito User Pool
cognitoUserPoolCli entProps?	`cognito.UserPoolCl ientProps`	Optional user provided props to override the default props for Cognito User Pool Client
logGroupProps?	`logs.LogGroupProps`	User provided props to override the default props for for the CloudWatchLogs LogGroup.

Pattern Properties

Name	Type	Description
userPool	`cognito.UserPool`	Returns an instance of cognito.UserPool created by the construct
userPoolClient	`cognito.UserPoolClient`	Returns an instance of cognito.UserPoolClient created by the construct
apiGateway	`api.RestApi`	Returns an instance of api.RestA pi created by the construct

Name	Type	Description
apiGatewayCloudWatchRole?	iam.Role	Returns an instance of the iam.Role created by the construct for API Gateway for CloudWatch access.
apiGatewayLogGroup	logs.LogGroup	Returns an instance of the LogGroup created by the construct for API Gateway access logging to CloudWatch.
apiGatewayAuthorizer	api.CfnAuthorizer	Returns an instance of the api.CfnAuthorizer created by the construct for API Gateway methods authorization.
lambdaFunction	lambda.Function	Returns an instance of lambda.Function created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon Cognito

- Set password policy for User Pools
- Enforce the advanced security mode for User Pools

Amazon API Gateway

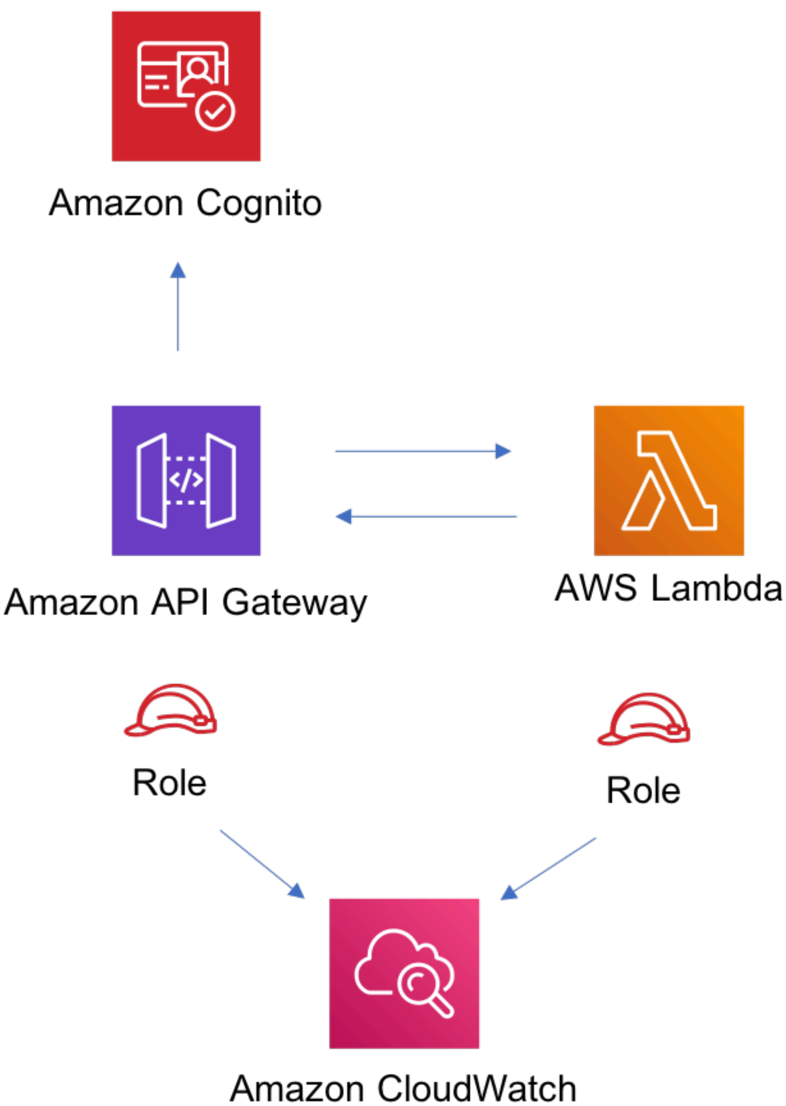
- Deploy an edge-optimized API endpoint
- Enable CloudWatch logging for API Gateway
- Configure least privilege access IAM role for API Gateway
- Set the default authorizationType for all API methods to Cognito User Pool

- Enable X-Ray Tracing

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-cognito-apigateway-lambda]()

aws-constructs-factories

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_constructs_factories`
Typescript	`@aws-solutions-constructs/aws-constructs-factories`
Java	`software.amazon.awsconstructs.services.constructsfactories`

Overview

This AWS Solutions Construct exposes the same code used to create our underlying resources as factories, so clients can create individual resources that are well-architected.

S3 Buckets

Create fully well-architected S3 buckets with as little as one function call. Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { ConstructsFactories } from '@aws-solutions-constructs/aws-constructs-
factories';

const factories = new ConstructsFactories(this, 'integ-test');

factories.s3BucketFactory('test', {});
```

Python

```
# TBD
```

Java

```
// TBD
```

S3BucketFactory Function Signature

```
s3BucketFactory(id: string, props: S3BucketFactoryProps): S3BucketFactoryResponse
```

S3BucketFactoryProps

Name	Type	Description
bucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the S3 Bucket.
logS3AccessLogs?	`boolean`	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with

Name	Type	Description
		associated storage costs for the logs. Enabling Access Logging is a best practice. default - true
loggingBucketProps?	s3.BucketProps	Optional user provided props to override the default props for the S3 Logging Bucket.

S3BucketFactoryResponse

Name	Type	Description
s3Bucket	s3.Bucket	The s3.Bucket created by the factory.
s3LoggingBucket?	s3.Bucket	The s3.Bucket created by the construct as the logging bucket for the primary bucket. If the logS3AccessLogs property is false, this value will be undefined.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

- An S3 Content Bucket
  - AWS managed Server Side Encryption (AES256)
  - Lifecycle rule to transition objects to Glacier storage class in 90 days
  - Access Logging enabled
  - All Public access blocked
  - Versioning enabled
  - UpdateReplacePolicy is delete

  - Deletion policy is delete
  - Bucket policy requiring SecureTransport
- An S3 Bucket for Access Logs
  - AWS managed Server Side Encryption (AES256)
  - All public access blocked
  - Versioning enabled
  - UpdateReplacePolicy is delete
  - Deletion policy is delete
  - Bucket policy requiring SecureTransport
  - Bucket policy granting PutObject privileges to the S3 logging service, from the content bucket in the content bucket account.
  - cfn_nag suppression of access logging finding (not logging access to the access log bucket)

Architecture

GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-constructs-factories](#)

aws-dynamodbstreams-lambda-elasticsearch-kibana

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_dynamodbstreams_elasticsearch_kibana`
Typescript	`@aws-solutions-constructs/aws-dynamodbstreams-lambda-elasticsearch-kibana`
Java	`software.amazon.awsconstructs.services.dynamodbstreamslambdaelasticsearchkibana`

Overview

This AWS Solutions Construct implements Amazon DynamoDB table with stream, AWS Lambda function and Amazon Elasticsearch Service with the least privileged permissions.

Some cluster configurations (e.g VPC access) require the existence of the `AWSServiceRoleForAmazonElasticsearchService` Service-Linked Role in your account.

You will need to create the service-linked role using the AWS CLI once in any account using this construct (it may have already been run to support other stacks):

```
aws iam create-service-linked-role --aws-service-name es.amazonaws.com
```

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps, Aws } from 'aws-cdk-lib';
import { DynamoDBStreamsToLambdaToElasticSearchAndKibana,
 DynamoDBStreamsToLambdaToElasticSearchAndKibanaProps } from '@aws-solutions-
constructs/aws-dynamodbstreams-lambda-elasticsearch-kibana';
import * as lambda from 'aws-cdk-lib/aws-lambda';

const constructProps: DynamoDBStreamsToLambdaToElasticSearchAndKibanaProps = {
  lambdaFunctionProps: {
    code: lambda.Code.fromAsset(`lambda`),
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler'
  },
  domainName: 'test-domain',
  // TODO: Ensure the Cognito domain name is globally unique
  cognitoDomainName: 'globallyuniquedomain' + Aws.ACCOUNT_ID
};

new DynamoDBStreamsToLambdaToElasticSearchAndKibana(this, 'test-dynamodbstreams-
lambda-elasticsearch-kibana', constructProps);
```

Python

```
from aws_solutions_constructs.aws_dynamodbstreams_lambda_elasticsearch_kibana
 import DynamoDBStreamsToLambdaToElasticSearchAndKibana,
 DynamoDBStreamsToLambdaToElasticSearchAndKibanaProps
from aws_cdk import (
    Stack,
    aws_lambda as _lambda,
    Aws,
)
from constructs import Construct

DynamoDBStreamsToLambdaToElasticSearchAndKibana(
    self, 'test-dynamodbstreams-lambda-elasticsearch-kibana',
    lambda_function_props=_lambda.FunctionProps(
        code=_lambda.Code.from_asset('lambda'),
        runtime=_lambda.Runtime.PYTHON_3_9,
        handler='index.handler'
```

```
    ),
    domain_name='test-domain',
    # TODO: Ensure the Cognito domain name is globally unique
    cognito_domain_name='globallyuniquedomain' + Aws.ACCOUNT_ID)
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Aws;
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import
 software.amazon.awsconstructs.services.dynamodbstreamslambdaelasticsearchkibana.*;

new DynamoDBStreamsToLambdaToElasticSearchAndKibana(this, "test-dynamodb-stream-
lambda-elasticsearch-kibana",
        new DynamoDBStreamsToLambdaToElasticSearchAndKibanaProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_16_X)
                        .code(Code.fromAsset("lambda"))
                        .handler("index.handler")
                        .build())
                .domainName("test-domain")
                .cognitoDomainName("globallyuniquedomain" + Aws.ACCOUNT_ID)
                .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.

Name	Type	Description
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
dynamoTableProps?	`dynamodb.TableProps`	Optional user provided props to override the default props for DynamoDB Table
existingTableInterface?	`dynamodb.ITable`	Existing instance of DynamoDB table object or interface , providing both this and `dynamoTableProps` will cause an error.
dynamoEventSourceProps?	`aws-lambda-event-sources.DynamoEventSourceProps`	Optional user provided props to override the default props for DynamoDB Event Source
esDomainProps?	`elasticsearch.CfnDomainProps`	Optional user provided props to override the default props for the Elasticsearch Service
domainName	`string`	Domain name for the Cognito and the Elasticsearch Service
cognitoDomainName?	`string`	Optional Cognito Domain Name, if provided it will be used for Cognito Domain, and domainName will be used for the Elasticsearch Domain.

Name	Type	Description
deploySqsDlqQueue?	`boolean`	Whether to deploy a SQS dead letter queue when a data record reaches the Maximum Retry Attempts or Maximum Record Age, its metadata like shard ID and stream ARN will be sent to an SQS queue.
sqsDlqQueueProps?	[`sqs.QueueProps`]()	Optional user provided properties for the SQS dead letter queue
createCloudWatchAlarms?	`boolean`	Whether to create recommend ed CloudWatch alarms
existingVpc?	[`ec2.IVpc`]()	An existing VPC in which to deploy the construct. Providing both this and `vpcProps` is an error.
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:

Name	Type	Description
vpcProps?	`ec2.VpcProps`	Optional user-provided properties to override the default properties for the new VPC. `enableDnsHostnames`, `enableDnsSupport`, `natGateways` and `subnetConfiguration` are set by the Construct, so any values for those properties supplied here will be overridden. If `deployVpc?` is not `true` then this property will be ignored.

Pattern Properties

Name	Type	Description
dynamoTableInterface	`dynamodb.ITable`	Returns an instance of dynamodb.ITable created by the construct
dynamoTable?	`dynamodb.Table`	Returns an instance of dynamodb.Table created by the construct. IMPORTANT: If existingTableInterface was provided in Pattern Construct Props, this property will be `undefined`
lambdaFunction	`lambda.Function`	Returns an instance of lambda.Function created by the construct

Name	Type	Description
userPool	`cognito.UserPool`	Returns an instance of cognito.UserPool created by the construct
userPoolClient	`cognito.UserPoolClient`	Returns an instance of cognito.UserPoolClient created by the construct
identityPool	`cognito.CfnIdentityPool`	Returns an instance of cognito.CfnIdentityPool created by the construct
elasticsearchDomain	`elasticsearch.CfnDomain`	Returns an instance of elasticsearch.CfnDomain created by the construct
elasticsearchDomain	`iam.Role`	Returns an instance of iam.Role created by the construct for elasticsearch.CfnDomain
cloudwatchAlarms?	`cloudwatch.Alarm[]`	Returns a list of cloudwatch.Alarm created by the construct
vpc?	`ec2.IVpc`	Returns an instance of the VPC created by the pattern, if `deployVpc?` is `true`, or `existingVpc?` is provided.

Lambda Function

This pattern requires a lambda function that can post data into the Elasticsearch from DynamoDB stream. A sample function is provided here.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon DynamoDB Table

- Set the billing mode for DynamoDB Table to On-Demand (Pay per request)
- Enable server-side encryption for DynamoDB Table using AWS managed KMS Key
- Creates a partition key called "id" for DynamoDB Table
- Retain the Table when deleting the CloudFormation stack
- Enable continuous backups and point-in-time recovery

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Enable Failure-Handling features like enable bisect on function Error, set defaults for Maximum Record Age (24 hours) & Maximum Retry Attempts (500) and deploy SQS dead-letter queue as destination on failure
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

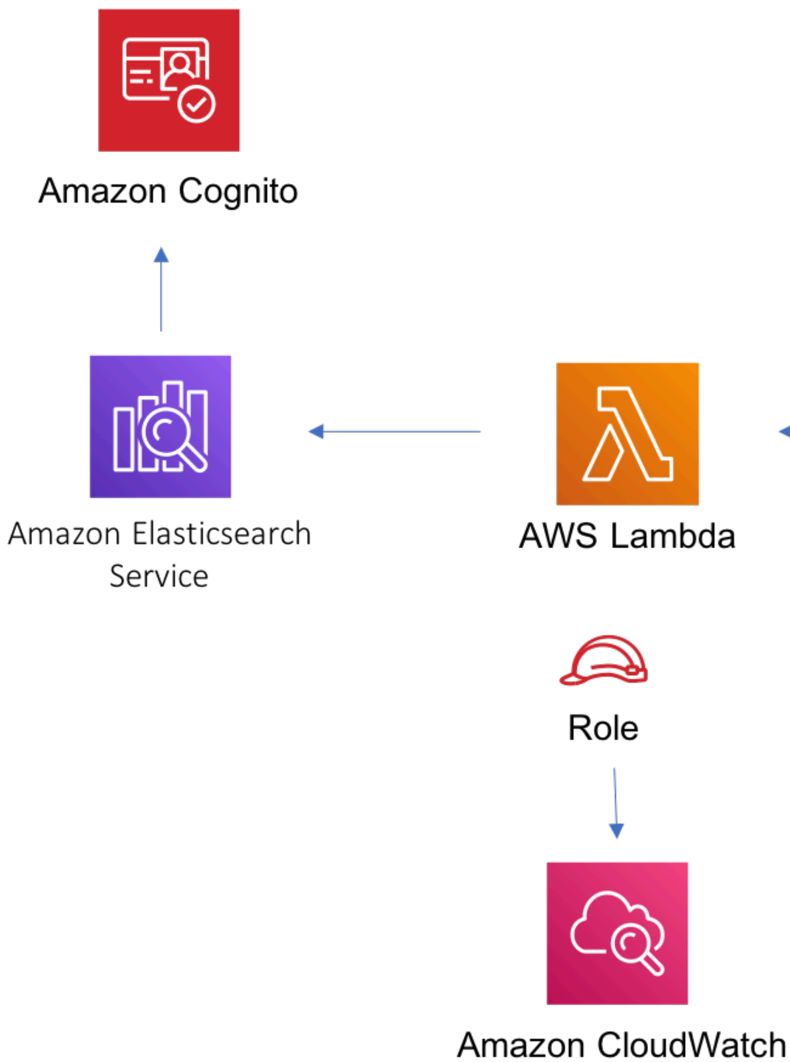
Amazon Cognito

- Set password policy for User Pools
- Enforce the advanced security mode for User Pools

Amazon Elasticsearch Service

- Deploy best practices CloudWatch Alarms for the Elasticsearch Domain
- Secure the Kibana dashboard access with Cognito User Pools
- Enable server-side encryption for Elasticsearch Domain using AWS managed KMS Key
- Enable node-to-node encryption for Elasticsearch Domain
- Configure the cluster for the Amazon ES domain

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-dynamodbstreams-lambda-elasticsearch-kibana

aws-dynamodbstreams-lambda

STABILITY EXPERIMENTAL

All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_dynamodbstreams_lambda
Typescript	@aws-solutions-constructs/aws-dynamodbstreams-lambda
Java	software.amazon.awsconstructs.services.dynamodbstreamslambda

Overview

This AWS Solutions Construct implements a pattern Amazon DynamoDB table with stream to invoke the AWS Lambda function with the least privileged permissions.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { DynamoDBStreamsToLambdaProps,  DynamoDBStreamsToLambda} from '@aws-solutions-constructs/aws-dynamodbstreams-lambda';
import * as lambda from 'aws-cdk-lib/aws-lambda';

new DynamoDBStreamsToLambda(this, 'test-dynamodbstreams-lambda', {
  lambdaFunctionProps: {
      code: lambda.Code.fromAsset(`lambda`),
      runtime: lambda.Runtime.NODEJS_16_X,
      handler: 'index.handler'
  },
});
```

Python

```
from aws_solutions_constructs.aws_dynamodbstreams_lambda import
 DynamoDBStreamsToLambda
from aws_cdk import (
  aws_lambda as _lambda,
  Stack
)
from constructs import Construct

DynamoDBStreamsToLambda(self, 'test-dynamodbstreams-lambda',
                        lambda_function_props=_lambda.FunctionProps(
                            code=_lambda.Code.from_asset('lambda'),
                            runtime=_lambda.Runtime.PYTHON_3_9,
                            handler='index.handler'
                        )
                        )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.dynamodbstreamslambda.*;

new DynamoDBStreamsToLambda(this, "test-dynamodbstreams-lambda",
        new DynamoDBStreamsToLambdaProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_16_X)
                        .code(Code.fromAsset("lambda"))
                        .handler("index.handler")
                        .build())
                .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
dynamoTableProps?	`dynamodb.TableProps`	Optional user provided props to override the default props for DynamoDB Table
existingTableInterface?	`dynamodb.ITable`	Existing instance of DynamoDB table object or interface , providing both this and `dynamoTableProps` will cause an error.
dynamoEventSourceProps?	`aws-lambda-event-sources.DynamoEventSourceProps`	Optional user provided props to override the default props for DynamoDB Event Source

Pattern Properties

Name	Type	Description
dynamoTableInterface	`dynamodb.ITable`	Returns an instance of dynamodb.ITable created by the construct

Name	Type	Description
dynamoTable?	`dynamodb.Table`	Returns an instance of dynamodb.Table created by the construct. IMPORTANT: If existingTableInterface was provided in Pattern Construct Props, this property will be `undefined`
lambdaFunction	`lambda.Function`	Returns an instance of lambda.Function created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

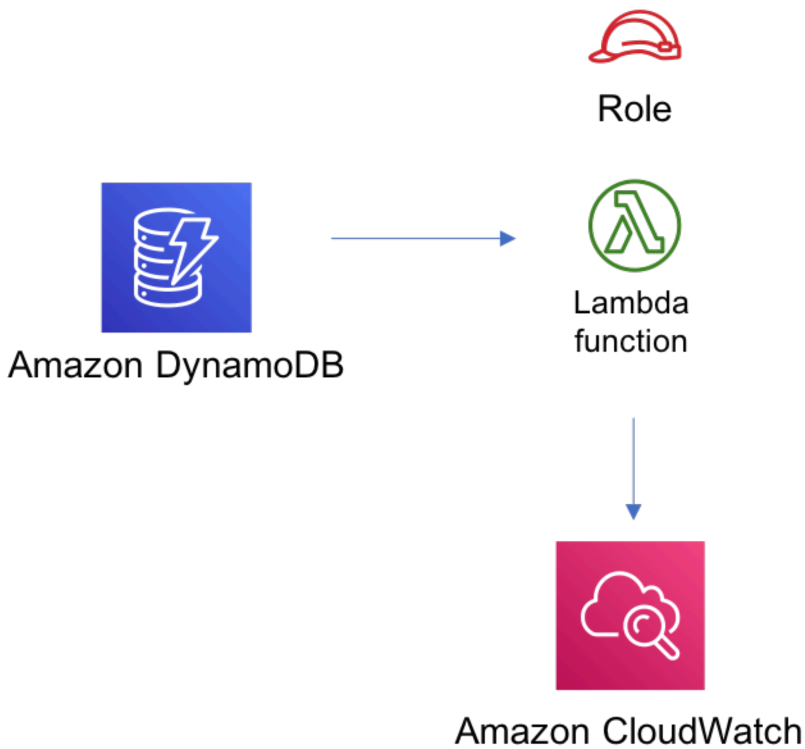
Amazon DynamoDB Table

- Set the billing mode for DynamoDB Table to On-Demand (Pay per request)
- Enable server-side encryption for DynamoDB Table using AWS managed KMS Key
- Creates a partition key called "id" for DynamoDB Table
- Retain the Table when deleting the CloudFormation stack
- Enable continuous backups and point-in-time recovery

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Enable Failure-Handling features like enable bisect on function Error, set defaults for Maximum Record Age (24 hours) & Maximum Retry Attempts (500) and deploy SQS dead-letter queue as destination on failure
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-dynamodbstreams-lambda

aws-eventbridge-kinesisfirehose-s3

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_eventbridge_kinesisfirehose_s3`

Language	Package
Typescript	`@aws-solutions-constructs/aws-eventbridge-kinesisfirehose-s3`
Java	`software.amazon.awsconstructs.services.eventbridgekinesisfirehoses3`

Overview

This AWS Solutions Construct implements an Amazon EventBridge Rule to send data to an Amazon Kinesis Data Firehose delivery stream connected to an Amazon S3 bucket.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps, Duration } from 'aws-cdk-lib';
import { EventbridgeToKinesisFirehoseToS3, EventbridgeToKinesisFirehoseToS3Props }
 from '@aws-solutions-constructs/aws-eventbridge-kinesisfirehose-s3';
import * as events from 'aws-cdk-lib/aws-events';

const EventbridgeToKinesisFirehoseToS3Props: EventbridgeToKinesisFirehoseToS3Props =
 {
  eventRuleProps: {
    schedule: events.Schedule.rate(Duration.minutes(5))
  }
};

new EventbridgeToKinesisFirehoseToS3(this, 'test-eventbridge-firehose-s3',
 EventbridgeToKinesisFirehoseToS3Props);
```

Python

```
from aws_solutions_constructs.aws_eventbridge_kinesis_firehose_s3 import
 EventbridgeToKinesisFirehoseToS3, EventbridgeToKinesisFirehoseToS3Props
from aws_cdk import (
```

```
    aws_events as events,
    Duration,
    Stack
)
from constructs import Construct

EventbridgeToKinesisFirehoseToS3(self, 'test-eventbridge-firehose-s3',
                                event_rule_props=events.RuleProps(
                                    schedule=events.Schedule.rate(
                                        Duration.minutes(5))
                                ))
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.Duration;
import software.amazon.awscdk.services.events.*;
import software.amazon.awsconstructs.services.eventbridgekinesisfirehoses3.*;

new EventbridgeToKinesisFirehoseToS3(this, "test-eventbridge-firehose-s3",
        new EventbridgeToKinesisFirehoseToS3Props.Builder()
                .eventRuleProps(new RuleProps.Builder()
                        .schedule(Schedule.rate(Duration.minutes(5)))
                        .build())
                .build());
```

Pattern Construct Props

Name	Type	Description
existingEventBusInterface?	events.IEventBus	Optional user-provided custom EventBus for construct to use. Providing both this and `eventBusProps` results an error.

Name	Type	Description
eventBusProps?	`events.EventBusProps`	Optional user-provided properties to override the default properties when creating a custom EventBus. Setting this value to `{}` will create a custom EventBus using all default properties. If neither this nor `existingEventBusInterface` is provided the construct will use the `default` EventBus. Providing both this and `existingEventBusInterface` results an error.
eventRuleProps	`events.RuleProps`	User provided eventRuleProps to override the defaults.
kinesisFirehoseProps?	`kinesisfirehose.CfnDeliveryStreamProps`	Optional user provided props to override the default props for Kinesis Firehose Delivery Stream
existingBucketObj?	`s3.IBucket`	Existing instance of S3 Bucket object. If this is provided, then also providing bucketProps is an error.
bucketProps?	`s3.BucketProps`	User provided props to override the default props for the S3 Bucket.
logGroupProps?	`logs.LogGroupProps`	User provided props to override the default props for for the CloudWatchLogs LogGroup.

Name	Type	Description
loggingBucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the S3 Logging Bucket.
logS3AccessLogs?	boolean	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with associated storage costs for the logs. Enabling Access Logging is a best practice. default - true

NOTE: `existingLoggingBucketObj` has been deprecated - to specify an existing Log Bucket, use `bucketProps.serverAccessLogsBucket`.

Pattern Properties

Name	Type	Description
eventBus?	`events.IEventBus`	Returns the instance of events.IEventBus used by the construct
eventsRule	`events.Rule`	Returns an instance of events.Rule created by the construct.
kinesisFirehose	`kinesisfirehose.CfnDeliveryStream`	Returns an instance of kinesisfirehose.CfnDeliveryStream created by the construct
s3Bucket?	`s3.Bucket`	Returns an instance of s3.Bucket created by the construct
s3LoggingBucket?	`s3.Bucket`	Returns an instance of s3.Bucket created by the

Name	Type	Description
		construct as the logging bucket for the primary bucket.
eventsRole	[`iam.Role`]()	Returns an instance of the iam.Role created by the construct for Events Rule
kinesisFirehoseRole	[`iam.Role`]()	Returns an instance of the iam.Role created by the construct for Kinesis Data Firehose delivery stream
kinesisFirehoseLogGroup	[`logs.LogGroup`]()	Returns an instance of the LogGroup created by the construct for Kinesis Data Firehose delivery stream
s3BucketInterface	[`s3.IBucket`]()	Returns an instance of s3.IBucke t created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon EventBridge Rule

- Configure least privilege access IAM role for Amazon EventBridge Rule to publish to the Kinesis Firehose Delivery Stream.

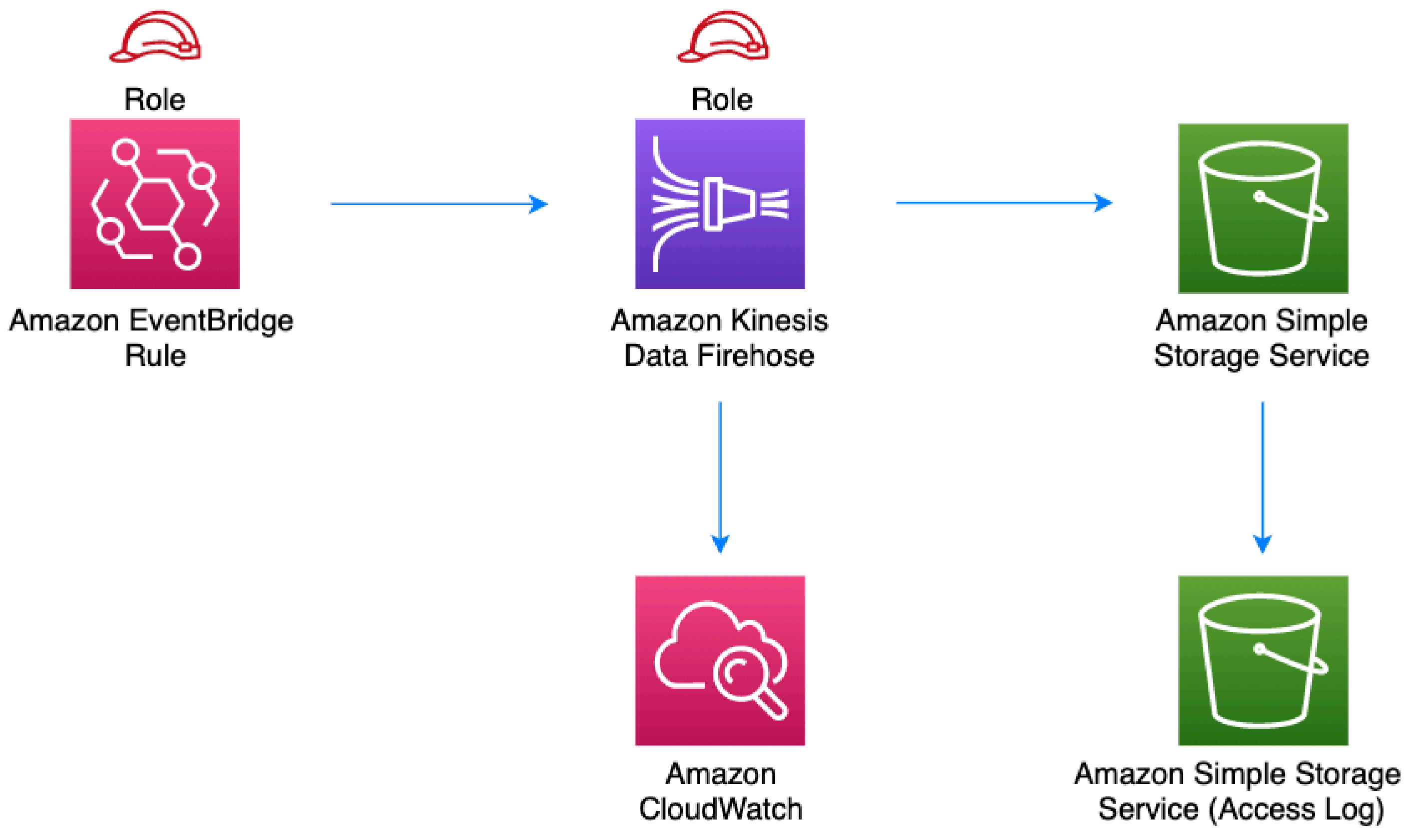
Amazon Kinesis Firehose

- Enable CloudWatch logging for Kinesis Firehose
- Configure least privilege access IAM role for Amazon Kinesis Firehose

Amazon S3 Bucket

- Configure Access logging for S3 Bucket
- Enable server-side encryption for S3 Bucket using AWS managed KMS Key
- Turn on the versioning for S3 Bucket
- Don't allow public access for S3 Bucket
- Retain the S3 Bucket when deleting the CloudFormation stack
- Applies Lifecycle rule to move noncurrent object versions to Glacier storage after 90 days

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:

[@aws-solutions-constructs/aws-eventbridge-kinesisfirehose-s3]

aws-eventbridge-kinesisstreams

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_eventbridge_kinesisstreams`
Typescript	`@aws-solutions-constructs/aws-eventbridge-kinesisstreams`
Java	`software.amazon.awsconstructs.services.eventbridgekinesisstreams`

Overview

This AWS Solutions Construct implements an Amazon EventBridge rule to send data to an Amazon Kinesis Data Stream

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps, Duration } from 'aws-cdk-lib';
import { EventbridgeToKinesisStreams, EventbridgeToKinesisStreamsProps } from "@aws-solutions-constructs/aws-eventbridge-kinesisstreams";
import * as events from 'aws-cdk-lib/aws-events';

const constructProps: EventbridgeToKinesisStreamsProps = {
  eventRuleProps: {
```

```
    schedule: events.Schedule.rate(Duration.minutes(5)),
  }
};

new EventbridgeToKinesisStreams(this, 'test-eventbridge-kinesis-streams',
 constructProps);
```

Python

```
from aws_solutions_constructs.aws_eventbridge_kinesis_streams import
 EventbridgeToKinesisStreams, EventbridgeToKinesisStreamsProps
from aws_cdk import (
    aws_events as events,
    Duration,
    Stack
)
from constructs import Construct

EventbridgeToKinesisStreams(self, 'test-eventbridge-kinesis-streams',
    event_rule_props=events.RuleProps(
        schedule=events.Schedule.rate(Duration.minutes(5)),
    ))
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.Duration;
import software.amazon.awscdk.services.events.*;
import software.amazon.awsconstructs.services.eventbridgekinesisstreams.*;

new EventbridgeToKinesisStreams(this, "test-eventbridge-kinesis-streams",
        new EventbridgeToKinesisStreamsProps.Builder()
                .eventRuleProps(new RuleProps.Builder()
                        .schedule(Schedule.rate(Duration.minutes(5)))
                        .build())
                .build());
```

Pattern Construct Props

Name	Type	Description
existingEventBusInterface?	`events.IEventBus`	Optional user-provided custom EventBus for construct to use. Providing both this and `eventBusProps` results an error.
eventBusProps?	`events.EventBusProps`	Optional user-provided properties to override the default properties when creating a custom EventBus. Setting this value to `{}` will create a custom EventBus using all default properties. If neither this nor `existingEventBusInterface` is provided the construct will use the `default` EventBus. Providing both this and `existingEventBusInterface` results an error.
eventRuleProps	`events.RuleProps`	User provided eventRuleProps to override the defaults.
existingStreamObj?	`kinesis.Stream`	Existing instance of Kinesis Stream, providing both this and `kinesisStreamProps` will cause an error.
kinesisStreamProps?	`kinesis.StreamProps`	Optional user-provided props to override the default props for the Kinesis stream.

Name	Type	Description
createCloudWatchAlarms	`boolean`	Whether to create recommend ed CloudWatch alarms.

Pattern Properties

Name	Type	Description
eventBus?	`events.IEventBus`	Returns the instance of events.IEventBus used by the construct
eventsRule	`events.Rule`	Returns an instance of events.Rule created by the construct.
kinesisStream	`kinesis.Stream`	Returns an instance of the Kinesis stream created by the pattern.
eventsRole?	`iam.Role`	Returns an instance of the iam.Role created by the construct for events rule.
cloudwatchAlarms?	`cloudwatch.Alarm[]`	Returns an instance of the cloudwatch.Alarm[] created by the construct.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

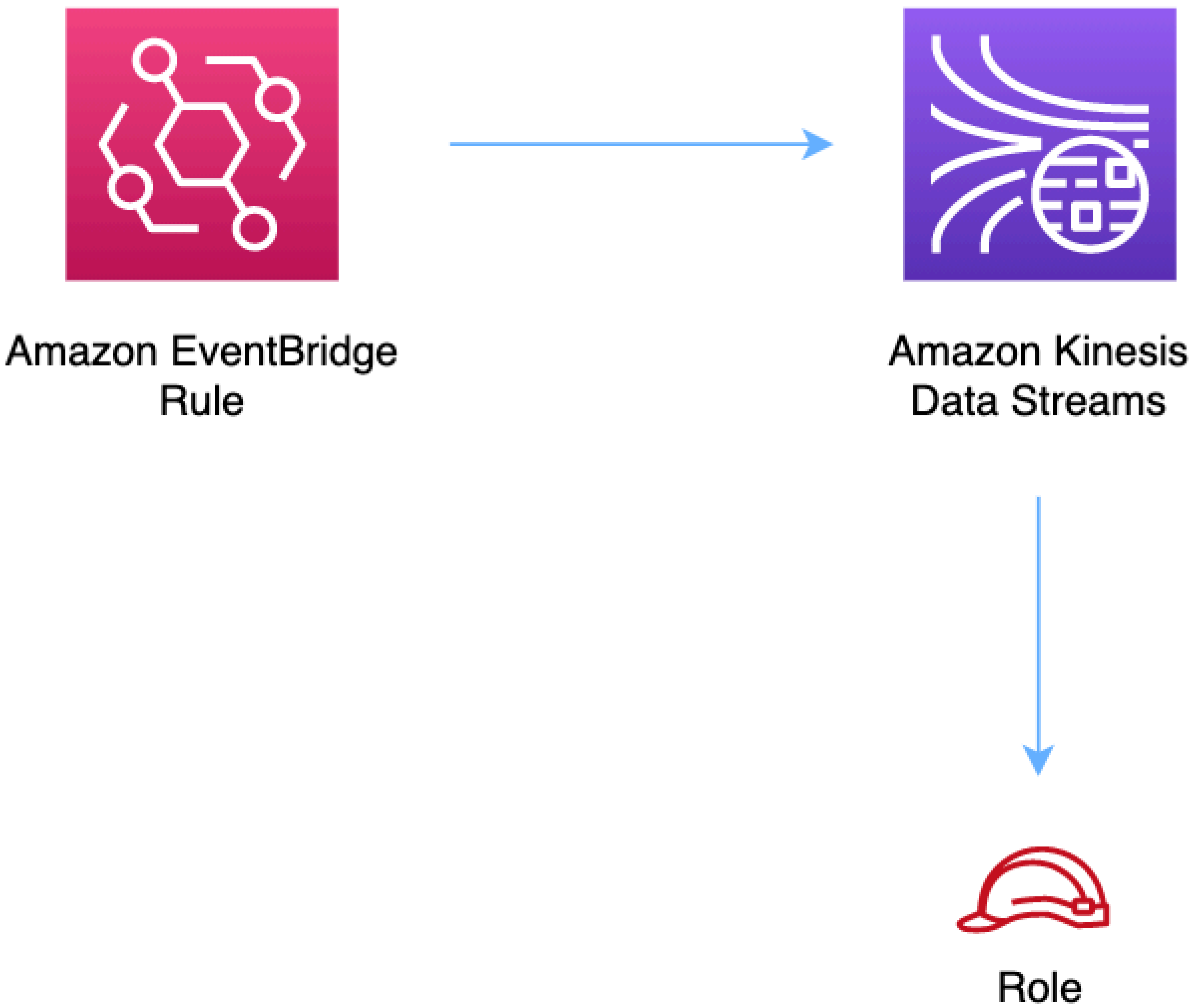
Amazon EventBridge Rule

- Configure least privilege access IAM role for EventBridge Rule to publish to the Kinesis Data Stream.

Amazon Kinesis Stream

- Enable server-side encryption for Kinesis Data Stream using AWS Managed KMS Key.

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-eventbridge-kinesisstreams]()

aws-eventbridge-lambda

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_eventbridge_lambda`
Typescript	`@aws-solutions-constructs/aws-eventbridge-lambda`
Java	`software.amazon.awsconstructs.services.eventbridgelambda`

Overview

This AWS Solutions Construct implements an AWS EventBridge rule and an AWS Lambda function.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps, Duration } from 'aws-cdk-lib';
import { EventbridgeToLambdaProps, EventbridgeToLambda } from '@aws-solutions-constructs/aws-eventbridge-lambda';
import * as lambda from 'aws-cdk-lib/aws-lambda';
import * as events from 'aws-cdk-lib/aws-events';
```

```
const constructProps: EventbridgeToLambdaProps = {
  lambdaFunctionProps: {
    code: lambda.Code.fromAsset(`lambda`),
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler'
  },
  eventRuleProps: {
    schedule: events.Schedule.rate(Duration.minutes(5))
  }
};

new EventbridgeToLambda(this, 'test-eventbridge-lambda', constructProps);
```

Python

```
from aws_solutions_constructs.aws_eventbridge_lambda import EventbridgeToLambda,
 EventbridgeToLambdaProps
from aws_cdk import (
    aws_lambda as _lambda,
    aws_events as events,
    Duration,
    Stack
)
from constructs import Construct

EventbridgeToLambda(self, 'test-eventbridge-lambda',
                    lambda_function_props=_lambda.FunctionProps(
                        code=_lambda.Code.from_asset('lambda'),
                        runtime=_lambda.Runtime.PYTHON_3_9,
                        handler='index.handler'
                    ),
                    event_rule_props=events.RuleProps(
                        schedule=events.Schedule.rate(
                            Duration.minutes(5))
                    ))
```

Java

```
import software.constructs.Construct;
```

```
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.Duration;
import software.amazon.awscdk.services.events.*;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.eventbridgelambda.*;

new EventbridgeToLambda(this, "test-eventbridge-lambda",
        new EventbridgeToLambdaProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_16_X)
                        .code(Code.fromAsset("lambda"))
                        .handler("index.handler")
                        .build())
                .eventRuleProps(new RuleProps.Builder()
                        .schedule(Schedule.rate(Duration.minutes(5)))
                        .build())
                .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
existingEventBusInterface?	`events.IEventBus`	Optional user-provided custom EventBus for construct to use. Providing both this and `eventBusProps` results an error.

Name	Type	Description
eventBusProps?	`events.EventBusProps`	Optional user-provided properties to override the default properties when creating a custom EventBus. Setting this value to `{}` will create a custom EventBus using all default properties. If neither this nor `existingEventBusInterface` is provided the construct will use the `default` EventBus. Providing both this and `existingEventBusInterface` results an error.
eventRuleProps	`events.RuleProps`	User provided eventRuleProps to override the defaults

Pattern Properties

Name	Type	Description
eventBus?	`events.IEventBus`	Returns the instance of events.IEventBus used by the construct
eventsRule	`events.Rule`	Returns an instance of events.Rule created by the construct
lambdaFunction	`lambda.Function`	Returns an instance of lambda.Function created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

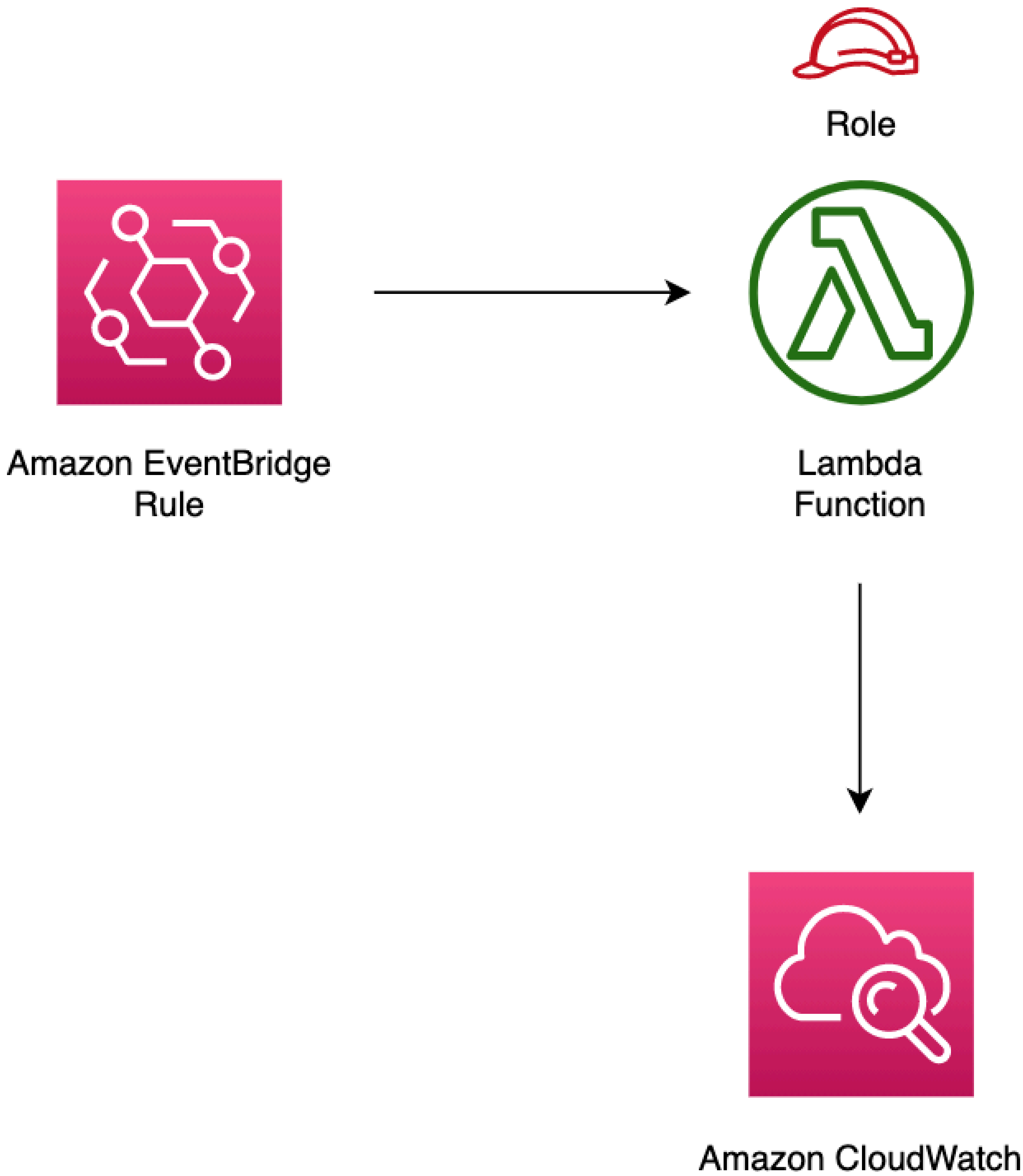
Amazon EventBridge Rule

- Grant least privilege permissions to EventBridge rule to trigger the Lambda Function

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
- AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-eventbridge-lambda]()

aws-eventbridge-sns

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_eventbridge_sns`
Typescript	`@aws-solutions-constructs/aws-eventbridge-sns`
Java	`software.amazon.awsconstructs.services.eventbridgesns`

Overview

This AWS Solutions Construct implements an AWS Events rule and an AWS SNS Topic.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps, Duration } from 'aws-cdk-lib';
import * as events from 'aws-cdk-lib/aws-events';
import * as iam from 'aws-cdk-lib/aws-iam';
import { EventbridgeToSnsProps, EventbridgeToSns } from "@aws-solutions-constructs/
aws-eventbridge-sns";
```

```
const constructProps: EventbridgeToSnsProps = {
  eventRuleProps: {
    schedule: events.Schedule.rate(Duration.minutes(5))
  }
};

const constructStack = new EventbridgeToSns(this, 'test-construct', constructProps);

// Grant yourself permissions to use the Customer Managed KMS Key
const policyStatement = new iam.PolicyStatement({
  actions: ["kms:Encrypt", "kms:Decrypt"],
  effect: iam.Effect.ALLOW,
  principals: [new iam.AccountRootPrincipal()],
  resources: ["*"]
});

constructStack.encryptionKey?.addToResourcePolicy(policyStatement);
```

Python

```
from aws_solutions_constructs.aws_eventbridge_sns import EventbridgeToSns,
 EventbridgeToSnsProps
from aws_cdk import (
    aws_events as events,
    aws_iam as iam,
    Duration,
    Stack
)
from constructs import Construct

construct_stack = EventbridgeToSns(self, 'test-construct',
                                    event_rule_props=events.RuleProps(
                                        schedule=events.Schedule.rate(
                                            Duration.minutes(5))
                                    ))

# Grant yourself permissions to use the Customer Managed KMS Key
policy_statement = iam.PolicyStatement(
    actions=["kms:Encrypt", "kms:Decrypt"],
    effect=iam.Effect.ALLOW,
    principals=[iam.AccountRootPrincipal()],
```

```
    resources=["*"]
)

construct_stack.encryption_key.add_to_resource_policy(policy_statement)
```

Java

```
import software.constructs.Construct;
import java.util.List;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.Duration;
import software.amazon.awscdk.services.events.*;
import software.amazon.awscdk.services.iam.*;
import software.amazon.awsconstructs.services.eventbridgesns.*;

final EventbridgeToSns constructStack = new EventbridgeToSns(this, "test-construct",
        new EventbridgeToSnsProps.Builder()
                .eventRuleProps(new RuleProps.Builder()
                        .schedule(Schedule.rate(Duration.minutes(5)))
                        .build())
                .build());

// Grant yourself permissions to use the Customer Managed KMS Key
final PolicyStatement policyStatement = PolicyStatement.Builder.create()
        .actions(List.of("kms:Encrypt", "kms:Decrypt"))
        .effect(Effect.ALLOW)
        .principals(List.of(new AccountRootPrincipal()))
        .resources(List.of("*"))
        .build();

constructStack.getEncryptionKey().addToResourcePolicy(policyStatement);
```

Pattern Construct Props

Name	Type	Description
eventRuleProps	[events.RuleProps](#)	User provided eventRuleProps to override the defaults.

Name	Type	Description
existingTopicObj?	sns.Topic	Existing instance of SNS Topic object, providing both this and `topicProps` will cause an error.
topicProps?	sns.TopicProps	User provided props to override the default props for the SNS Topic.
existingEventBusInterface?	events.IEventBus	Optional user-provided custom EventBus for construct to use. Providing both this and `eventBusProps` results an error.
eventBusProps?	events.EventBusProps	Optional user-provided properties to override the default properties when creating a custom EventBus. Setting this value to `{}` will create a custom EventBus using all default properties. If neither this nor `existingEventBusInterface` is provided the construct will use the `default` EventBus. Providing both this and `existingEventBusInterface` results an error.

Name	Type	Description
enableEncryptionWithCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the SNS Topic is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined: topicProps.masterKey, encryptionKey or encryptionKeyProps.
encryptionKey?	[`kms.Key`]()	An optional, imported encryption key to encrypt the SNS Topic with.
encryptionKeyProps?	[`kms.KeyProps`]()	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SNS Topic with.

Pattern Properties

Name	Type	Description
eventBus?	[`events.IEventBus`]()	Returns the instance of events.IEventBus used by the construct
eventsRule	[`events.Rule`]()	Returns an instance of events.Rule created by the construct
snsTopic	[`sns.Topic`]()	Returns an instance of sns.Topic created by the construct

Name	Type	Description
encryptionKey?	`kms.Key`	Returns an instance of kms Key used for the SNS Topic.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

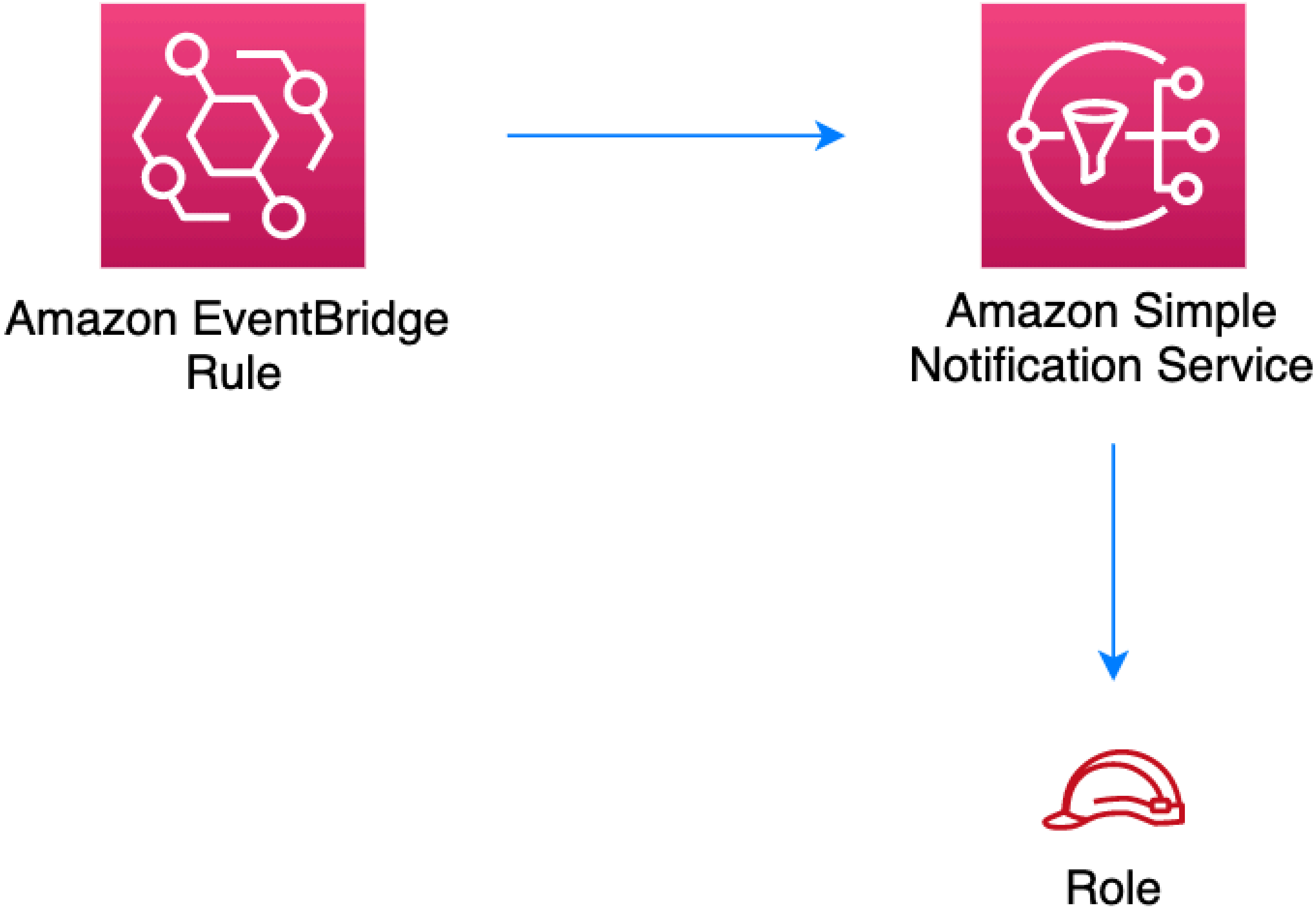
Amazon EventBridge Rule

- Grant least privilege permissions to EventBridge Rule to publish to the SNS Topic.

Amazon SNS Topic

- Configure least privilege access permissions for SNS Topic.
- Enable server-side encryption forSNS Topic using Customer managed KMS Key.
- Enforce encryption of data in transit.

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-eventbridge-sns

aws-eventbridge-sqs

CFN-RESOURCES STABLE

Language	Package
Python	aws_solutions_constructs.aws_eventbridge_sqs
Typescript	@aws-solutions-constructs/aws-eventbridge-sqs
Java	software.amazon.awsconstructs.services.eventbridgesqs

Overview

This AWS Solutions Construct implements an Amazon EventBridge rule and an AWS SQS Queue.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps, Duration } from 'aws-cdk-lib';
import * as events from 'aws-cdk-lib/aws-events';
import * as iam from 'aws-cdk-lib/aws-iam';
import { EventbridgeToSqsProps, EventbridgeToSqs } from "@aws-solutions-constructs/
aws-eventbridge-sqs";

const constructProps: EventbridgeToSqsProps = {
  eventRuleProps: {
    schedule: events.Schedule.rate(Duration.minutes(5))
  }
};

const constructStack = new EventbridgeToSqs(this, 'test-construct', constructProps);

// Grant yourself permissions to use the Customer Managed KMS Key
const policyStatement = new iam.PolicyStatement({
  actions: ["kms:Encrypt", "kms:Decrypt"],
  effect: iam.Effect.ALLOW,
  principals: [new iam.AccountRootPrincipal()],
  resources: ["*"]
```

```
});

constructStack.encryptionKey?.addToResourcePolicy(policyStatement);
```

Python

```
from aws_solutions_constructs.aws_eventbridge_sqs import EventbridgeToSqsProps,
 EventbridgeToSqs
from aws_cdk import (
    aws_events as events,
    aws_iam as iam,
    Duration,
    Stack
)
from constructs import Construct

construct_stack = EventbridgeToSqs(self, 'test-construct',
                                    event_rule_props=events.RuleProps(
                                        schedule=events.Schedule.rate(
                                            Duration.minutes(5))
                                    ))

# Grant yourself permissions to use the Customer Managed KMS Key
policy_statement = iam.PolicyStatement(
    actions=["kms:Encrypt", "kms:Decrypt"],
    effect=iam.Effect.ALLOW,
    principals=[iam.AccountRootPrincipal()],
    resources=["*"]
)

construct_stack.encryption_key.add_to_resource_policy(policy_statement)
```

Java

```
import software.constructs.Construct;
import java.util.List;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.Duration;
import software.amazon.awscdk.services.events.*;
```

```
import software.amazon.awscdk.services.iam.*;
import software.amazon.awsconstructs.services.eventbridgesqs.*;

final EventbridgeToSqs constructStack = new EventbridgeToSqs(this, "test-construct",
        new EventbridgeToSqsProps.Builder()
                .eventRuleProps(new RuleProps.Builder()
                        .schedule(Schedule.rate(Duration.minutes(5)))
                        .build())
                .build());

// Grant yourself permissions to use the Customer Managed KMS Key
final PolicyStatement policyStatement = PolicyStatement.Builder.create()
        .actions(List.of("kms:Encrypt", "kms:Decrypt"))
        .effect(Effect.ALLOW)
        .principals(List.of(new AccountRootPrincipal()))
        .resources(List.of("*"))
        .build();

constructStack.getEncryptionKey().addToResourcePolicy(policyStatement);
```

Pattern Construct Props

Name	Type	Description
existingEventBusInterface?	[events.IEventBus](#)	Optional user-provided custom EventBus for construct to use. Providing both this and `eventBusProps` results an error.
eventBusProps?	[events.EventBusProps](#)	Optional user-provided properties to override the default properties when creating a custom EventBus. Setting this value to {} will create a custom EventBus using all default properties. If neither this nor `existingEventBusInterface` is

Name	Type	Description
		provided the construct will use the `default` EventBus. Providing both this and `existingEventBusInterface` results an error.
eventRuleProps	[events.RuleProps](#)	User provided eventRuleProps to override the defaults.
existingQueueObj?	[sqs.Queue](#)	An optional, existing SQS queue to be used instead of the default queue. Providing both this and `queueProps` will cause an error.
queueProps?	[sqs.QueueProps](#)	User provided props to override the default props for the SQS Queue.
enableQueuePurging?	`boolean`	Whether to grant additional permissions to the Lambda function enabling it to purge the SQS queue. Defaults to `false`.
deployDeadLetterQueue?	`boolean`	Whether to create a secondary queue to be used as a dead letter queue. Defaults to `true`.
deadLetterQueueProps?	[sqs.QueueProps](#)	Optional user-provided props to override the default props for the dead letter queue. Only used if the `deployDeadLetterQueue` property is set to true.

Name	Type	Description
maxReceiveCount?	`number`	The number of times a message can be unsuccessfully dequeued before being moved to the dead letter queue. Defaults to 15.
enableEncryptionWithCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the queue is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined: queueProps.encryptionMasterKey, encryptionKey or encryptionKeyProps.
encryptionKey?	`kms.Key`	An optional, imported encryption key to encrypt the SQS Queue with.
encryptionKeyProps?	`kms.KeyProps`	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SQS queue with.

Pattern Properties

Name	Type	Description
eventBus?	`events.IEventBus`	Returns the instance of events.IEventBus used by the construct

Name	Type	Description
eventsRule	`events.Rule`	Returns an instance of events.Rule created by the construct
sqsQueue	`sqs.Queue`	Returns an instance of sqs.Queue created by the construct
encryptionKey?	`kms.Key`	Returns an instance of kms Key used for the SQS queue.
deadLetterQueue?	`sqs.Queue`	Returns an instance of the dead-letter SQS queue created by the pattern.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

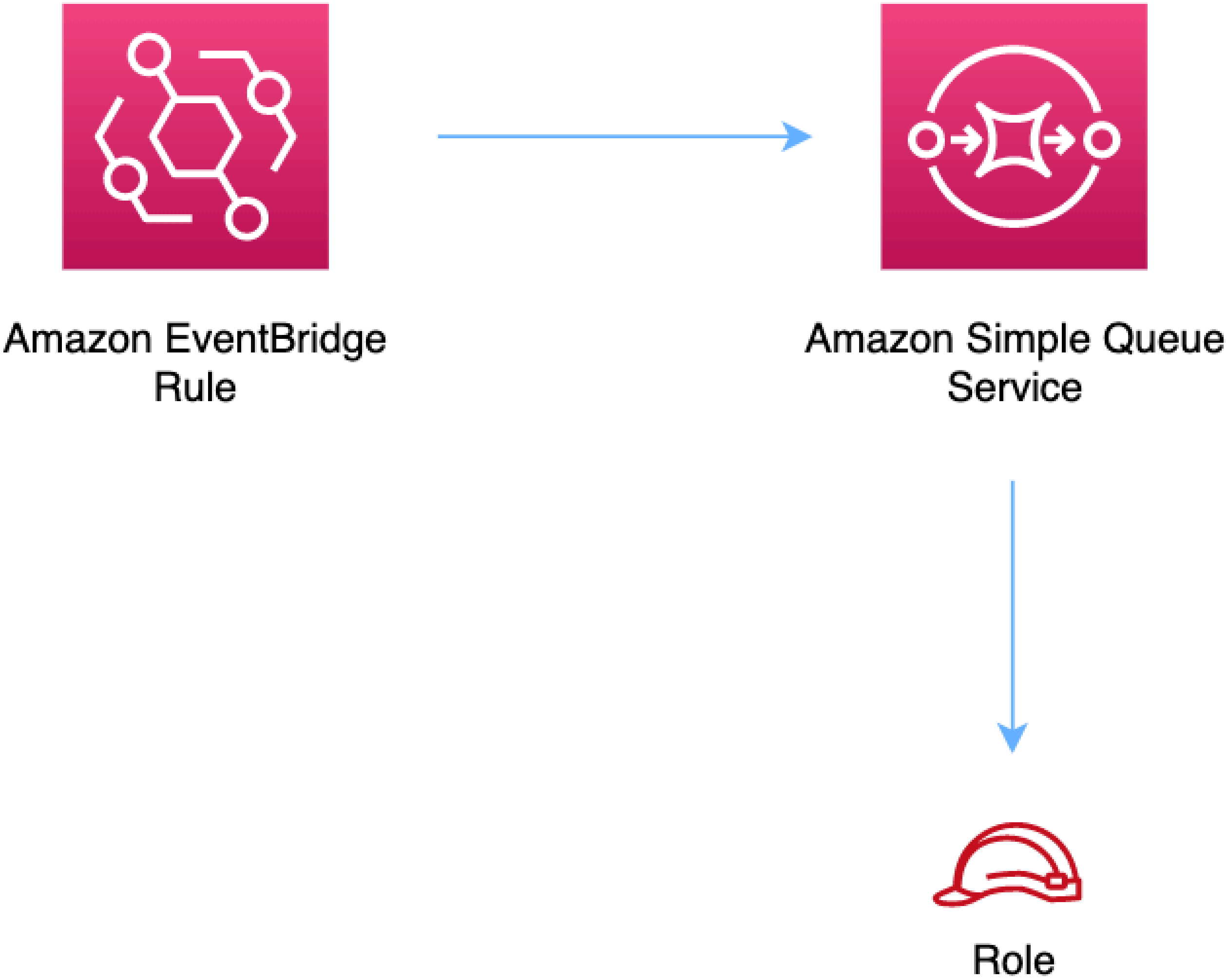
Amazon EventBridge Rule

- Grant least privilege permissions to EventBridge rule to publish to the SQS Queue.

Amazon SQS Queue

- Deploy SQS dead-letter queue for the source SQS Queue.
- Enable server-side encryption for source SQS Queue using Customer managed KMS Key.
- Enforce encryption of data in transit.

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-eventbridge-sqs

aws-eventbridge-stepfunctions

CFN-RESOURCES STABLE

Language	Package
Python	aws_solutions_constructs.aws_eventbridge_stepfunctions
Typescript	@aws-solutions-constructs/aws-eventbridge-stepfunctions
Java	software.amazon.awsconstructs.services.eventbridgestepfunctions

Overview

This AWS Solutions Construct implements an AWS Events rule and an AWS Step Functions State Machine

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps, Duration } from 'aws-cdk-lib';
import { EventbridgeToStepfunctions, EventbridgeToStepfunctionsProps } from '@aws-solutions-constructs/aws-eventbridge-stepfunctions';
import * as stepfunctions from 'aws-cdk-lib/aws-stepfunctions';
import * as events from 'aws-cdk-lib/aws-events';

const startState = new stepfunctions.Pass(this, 'StartState');

const constructProps: EventbridgeToStepfunctionsProps = {
  stateMachineProps: {
    definition: startState
  },
  eventRuleProps: {
    schedule: events.Schedule.rate(Duration.minutes(5))
  }
};
```

```
new EventbridgeToStepfunctions(this, 'test-eventbridge-stepfunctions-stack',
 constructProps);
```

Python

```
from aws_solutions_constructs.aws_eventbridge_stepfunctions import
 EventbridgeToStepfunctions, EventbridgeToStepfunctionsProps
from aws_cdk import (
    aws_stepfunctions as stepfunctions,
    aws_events as events,
    Duration,
    Stack
)
from constructs import Construct

startState = stepfunctions.Pass(self, 'StartState')

EventbridgeToStepfunctions(self, 'test-eventbridge-stepfunctions-stack',
                            state_machine_props=stepfunctions.StateMachineProps(
                                definition=startState
                            ),
                            event_rule_props=events.RuleProps(
                                schedule=events.Schedule.rate(
                                    Duration.minutes(5))
                            ))
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.Duration;
import software.amazon.awscdk.services.events.*;
import software.amazon.awscdk.services.stepfunctions.*;
import software.amazon.awsconstructs.services.eventbridgestepfunctions.*;

final Pass startState = new Pass(this, "StartState");

new EventbridgeToStepfunctions(this,
        "test-eventbridge-stepfunctions-stack",
```

```
        new EventbridgeToStepfunctionsProps.Builder()
                .stateMachineProps(new StateMachineProps.Builder()
                        .definition(startState)
                        .build())
                .eventRuleProps(new RuleProps.Builder()
                        .schedule(Schedule.rate(Duration.minutes(5)))
                        .build())
                .build());
```

Pattern Construct Props

Name	Type	Description
stateMachineProps	`sfn.StateMachineProps`	Optional user provided props to override the default props for sfn.StateMachine
existingEventBusInterface?	`events.IEventBus`	Optional user-provided custom EventBus for construct to use. Providing both this and `eventBusProps` results an error.
eventBusProps?	`events.EventBusProps`	Optional user-provided properties to override the default properties when creating a custom EventBus. Setting this value to `{}` will create a custom EventBus using all default properties. If neither this nor `existingEventBusInterface` is provided the construct will use the `default` EventBus. Providing both this and `existingEventBusInterface` results an error.

Name	Type	Description
eventRuleProps	events.RuleProps	User provided eventRuleProps to override the defaults
createCloudWatchAlarms	`boolean`	Whether to create recommend ed CloudWatch alarms
logGroupProps?	logs.LogGroupProps	User provided props to override the default props for for the CloudWatchLogs LogGroup.

Pattern Properties

Name	Type	Description
eventBus?	events.IEventBus	Returns the instance of events.IEventBus used by the construct
eventsRule	events.Rule	Returns an instance of events.Rule created by the construct
stateMachine	sfn.StateMachine	Returns an instance of sfn.State Machine created by the construct
stateMachineLogGroup	logs.ILogGroup	Returns an instance of the ILogGroup created by the construct for StateMachine
cloudwatchAlarms?	[cloudwatch.Alarm[]](cloudwatch.Alarm[])	Returns a list of cloudwatc h.Alarm created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

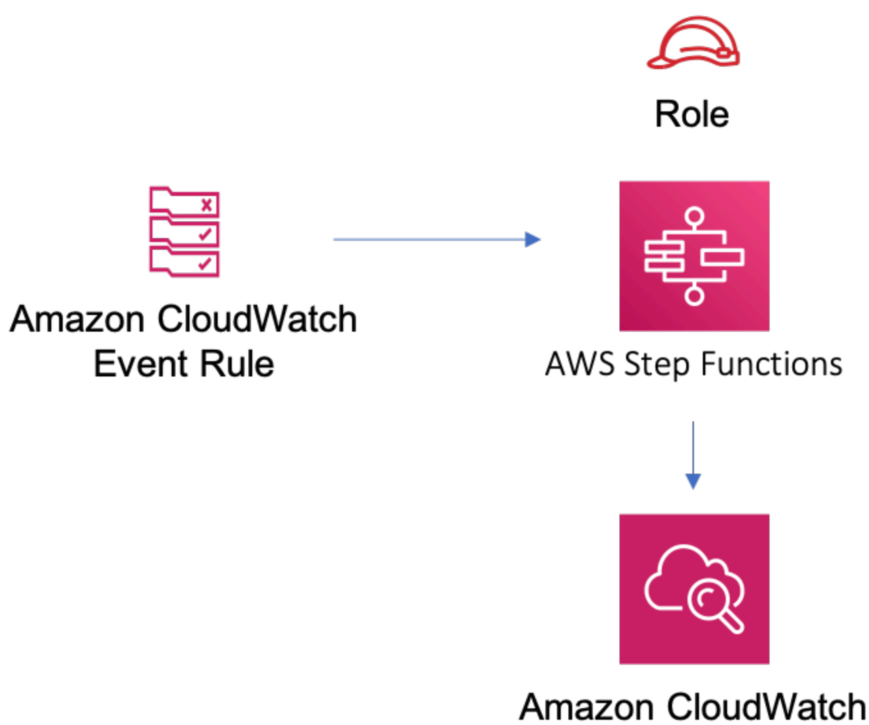
Amazon CloudWatch Events Rule

- Grant least privilege permissions to CloudWatch Events to trigger the Lambda Function

AWS Step Function

- Enable CloudWatch logging for API Gateway
- Deploy best practices CloudWatch Alarms for the Step Function

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-eventbridge-stepfunctions

aws-fargate-dynamodb

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_fargate_dynamodb`
Typescript	`@aws-solutions-constructs/aws-fargate-dynamodb`
Java	`software.amazon.awsconstructs.services.fargatedynamodb`

Overview

This AWS Solutions Construct implements an AWS Fargate service that can write/read to an Amazon DynamoDB table

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { FargateToDynamoDB, FargateToDynamoDBProps } from '@aws-solutions-
constructs/aws-fargate-dynamodb';

const constructProps: FargateToDynamoDBProps = {
  publicApi: true,
  ecrRepositoryArn: "arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-repo",
};

new FargateToDynamoDB(stack, 'test-construct', constructProps);
```

Python

```
from aws_solutions_constructs.aws_fargate_dynamodb import FargateToDynamoDB,
 FargateToDynamoDBProps
from aws_cdk import (
    Stack
)
from constructs import Construct

FargateToDynamoDB(self, 'test_construct',
            public_api=True,
            ecr_repository_arn="arn:aws:ecr:us-east-1:123456789012:repository/your-
ecr-repo")
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.fargatedynamodb.*;

new FargateToDynamoDB(this, "test-construct", new FargateToDynamoDBProps.Builder()
        .publicApi(true)
        .ecrRepositoryArn("arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-
repo")
        .build());
```

Pattern Construct Props

Name	Type	Description
publicApi	`boolean`	Whether the construct is deploying a private or public API. This has implications for the VPC.
vpcProps?	`ec2.VpcProps`	Optional custom properties for a VPC the construct will create. This VPC will be used by any Private Hosted Zone the construct creates (that's why loadBalancerProps and privateHostedZoneProps can't include a VPC). Providing both this and existingVpc is an error.
existingVpc?	`ec2.IVpc`	An existing VPC in which to deploy the construct. Providing both this and vpcProps is an error. If the client provides an existing load balancer and/or existing Private Hosted Zone, those constructs must exist in this VPC.
clusterProps?	`ecs.ClusterProps`	Optional properties to create a new ECS cluster. To provide an existing cluster, use the cluster attribute of fargateServiceProp s.
ecrRepositoryArn?	`string`	The arn of an ECR Repository containing the image to use to generate the containers. Either this or the image property

Name	Type	Description
		of containerDefinitionProps must be provided. format: arn:aws:ecr:*region*:*account number*:repository/*Repository Name*
ecrImageVersion?	`string`	The version of the image to use from the repository. Defaults to "Latest"
containerDefinitionProps?	[`ecs.ContainerDefinitionProps | any`]()	Optional props to define the container created for the Fargate Service (defaults found in fargate-defaults.ts)
fargateTaskDefinitionProps?	[`ecs.FargateTaskDefinitionProps | any`]()	Optional props to define the Fargate Task Definition for this construct (defaults found in fargate-defaults.ts)
fargateServiceProps?	[`ecs.FargateServiceProps | any`]()	Optional values to override default Fargate Task definition properties (fargate-defaults.ts). The construct will default to launching the service is the most isolated subnets available (precedence: Isolated, Private and Public). Override those and other defaults here.

Name	Type	Description
existingFargateSer viceObject?	ecs.FargateService	A Fargate Service already instantiated (probably by another Solutions Construct). If this is specified, then no props defining a new service can be provided, including: ecrImageVersion, container DefinitionProps, fargateTa skDefinitionProps, ecrReposi toryArn, fargateServiceProps, clusterProps
existingContainerD efinitionObject?	ecs.ContainerDefin ition	A container definition already instantiated as part of a Fargate service. This must be the container in the existingF argateServiceObject
dynamoTableProps?	dynamodb.TableProps	Optional user provided props to override the default props for DynamoDB Table.
existingTableInterface?	dynamodb.ITable	Existing instance of DynamoDB table object or interface , providing both this and `dynamoTableProps` will cause an error.
tablePermissions?	`string`	Optional table permissio ns to grant to the Fargate service. One of the following may be specified: `All`, `Read`, `ReadWrite` , `Write`.

Name	Type	Description
tableArnEnvironmentVariableName?	`string`	Optional Name for the container environment variable set to the ARN for the DynamoDB table. Default: DYNAMODB_TABLE_ARN
tableEnvironmentVariableName?	`string`	Optional Name for the container environment variable set to the DynamoDB table name. Default: DYNAMODB_TABLE_NAME

Pattern Properties

Name	Type	Description
vpc	`ec2.IVpc`	The VPC used by the construct (whether created by the construct or provided by the client)
service	`ecs.FargateService`	The AWS Fargate service used by this construct (whether created by this construct or passed to this construct at initialization)
container	`ecs.ContainerDefinition`	The container associated with the AWS Fargate service in the service property.
dynamoTableInterface	`dynamodb.ITable`	Returns an instance of `dynamodb.ITable` created by the construct or the interface

Name	Type	Description
		provided in existingTableInter face.
dynamoTable?	[`dynamodb.Table`]()	Returns an instance of `dynamodb.Table` created by the construct. IMPORTANT : If existingTableInterface was provided in Pattern Construct Props, this property will be `undefined` .

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

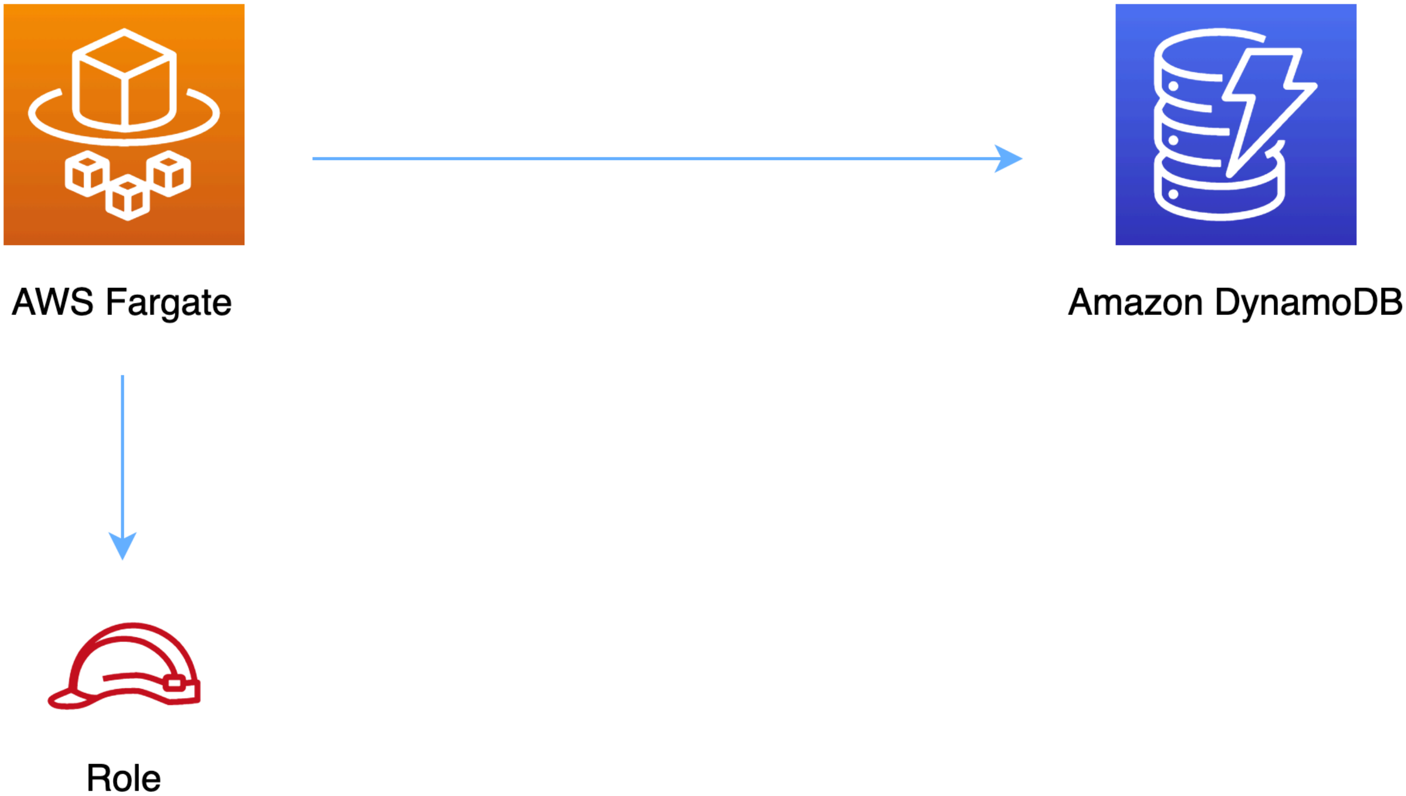
AWS Fargate Service

- Sets up an AWS Fargate service
  - Uses the existing service if provided
  - Creates a new service if none provided.
    - Service will run in isolated subnets if available, then private subnets if available and finally public subnets
  - Adds environment variables to the container with the ARN and Name of the DynamoDB table
  - Add permissions to the container IAM role allowing it to publish to the DynamoDB table

Amazon DynamoDB Table

- Sets up an Amazon DynamoDB table
  - Uses an existing table if one is provided, otherwise creates a new one
- Adds an Interface Endpoint to the VPC for DynamoDB (the service by default runs in Isolated or Private subnets)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-fargate-dynamodb

aws-fargate-eventbridge

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_fargate_eventbridge
Typescript	@aws-solutions-constructs/aws-fargate-eventbridge
Java	software.amazon.awsconstructs.services.fargateeventbridge

This AWS Solutions Construct implements an AWS Fargate service connected to an Amazon EventBridge.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { FargateToEventbridge, FargateToEventbridgeProps } from '@aws-solutions-
constructs/aws-fargate-eventbridge';

const constructProps: FargateToEventbridgeProps = {
  publicApi: true,
  ecrRepositoryArn: "arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-repo",
};

new FargateToEventbridge(this, 'test-construct', constructProps);
```

Python

```
from aws_solutions_constructs.aws_fargate_eventbridge import FargateToEventbridge,
 FargateToEventbridgeProps
from aws_cdk import (
    Stack
)
from constructs import Construct
```

```
FargateToEventbridge(self, 'test_construct',
            public_api=True,
            ecr_repository_arn="arn:aws:ecr:us-east-1:123456789012:repository/your-
ecr-repo")
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.fargateeventbridge.*;

new FargateToEventbridge(this, "test_construct", new
 FargateToEventbridgeProps.Builder()
        .publicApi(true)
        .ecrRepositoryArn("arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-
repo")
        .build());
```

Pattern Construct Props

Name	Type	Description
publicApi	boolean	Whether the construct is deploying a private or public API. This has implications for the VPC.
vpcProps?	ec2.VpcProps	Optional custom properties for a VPC the construct will create. This VPC will be used by any Private Hosted Zone the construct creates (that's why loadBalancerProps and privateHostedZoneProps can't

Name	Type	Description
		include a VPC). Providing both this and existingVpc is an error.
existingVpc?	[ec2.IVpc]()	An existing VPC in which to deploy the construct. Providing both this and vpcProps is an error. If the client provides an existing load balancer and/or existing Private Hosted Zone, those constructs must exist in this VPC.
clusterProps?	[ecs.ClusterProps]()	Optional properties to create a new ECS cluster. To provide an existing cluster, use the cluster attribute of fargateServiceProp s.
ecrRepositoryArn?	`string`	The arn of an ECR Repository containing the image to use to generate the containers. Either this or the image property of containerDefinitionProps must be provided. format: arn:aws:ecr:*region*:*account number*:repository/*Repository Name*
ecrImageVersion?	`string`	The version of the image to use from the repository. Defaults to "Latest"
containerDefinitionProps?	[ecs.ContainerDefinitionProps | any]()	Optional props to define the container created for the Fargate Service (defaults found in fargate-defaults.ts)

Name	Type	Description
fargateTaskDefinitionProps?	ecs.FargateTaskDefinitionProps | any	Optional props to define the Fargate Task Definition for this construct (defaults found in fargate-defaults.ts)
fargateServiceProps?	ecs.FargateServiceProps | any	Optional values to override default Fargate Task definition properties (fargate-defaults.ts). The construct will default to launching the service is the most isolated subnets available (precedence: Isolated, Private and Public). Override those and other defaults here.
existingFargateServiceObject?	ecs.FargateService	A Fargate Service already instantiated (probably by another Solutions Construct). If this is specified, then no props defining a new service can be provided, including: ecrImageVersion, containerDefinitionProps, fargateTaskDefinitionProps, ecrRepositoryArn, fargateServiceProps, clusterProps
existingContainerDefinitionObject?	ecs.ContainerDefinition	A container definition already instantiated as part of a Fargate service. This must be the container in the existingFargateServiceObject

Name	Type	Description
existingEventBusInterface?	[events.IEventBus](#)	Optional user-provided custom event bus for construct to use. Providing both this and `eventBusProps` results an error.
eventBusProps?	[events.EventBusProps](#)	Optional user-provided properties to override the default properties when creating a custom event bus. Setting this value to `{}` will create a custom event bus using all default properties. If neither this nor `existingEventBusInterface` is provided the construct will use the `default` event bus. Providing both this and `existingEventBusInterface` results an error.
eventBusEnvironmentVariableName?	`string`	Optional Name for the container environment variable set to the DynamoDB table name. Default: EVENTBUS_NAME

Pattern Properties

Name	Type	Description
vpc	[ec2.IVpc](#)	The VPC used by the construct (whether created by the

Name	Type	Description
		construct or provided by the client)
service	ecs.FargateService	The AWS Fargate service used by this construct (whether created by this construct or passed to this construct at initialization)
container	ecs.ContainerDefinition	The container associated with the AWS Fargate service in the service property.
eventBus?	`events.IEventBus`	Returns the instance of `events.IEventBus` used by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS Fargate Service

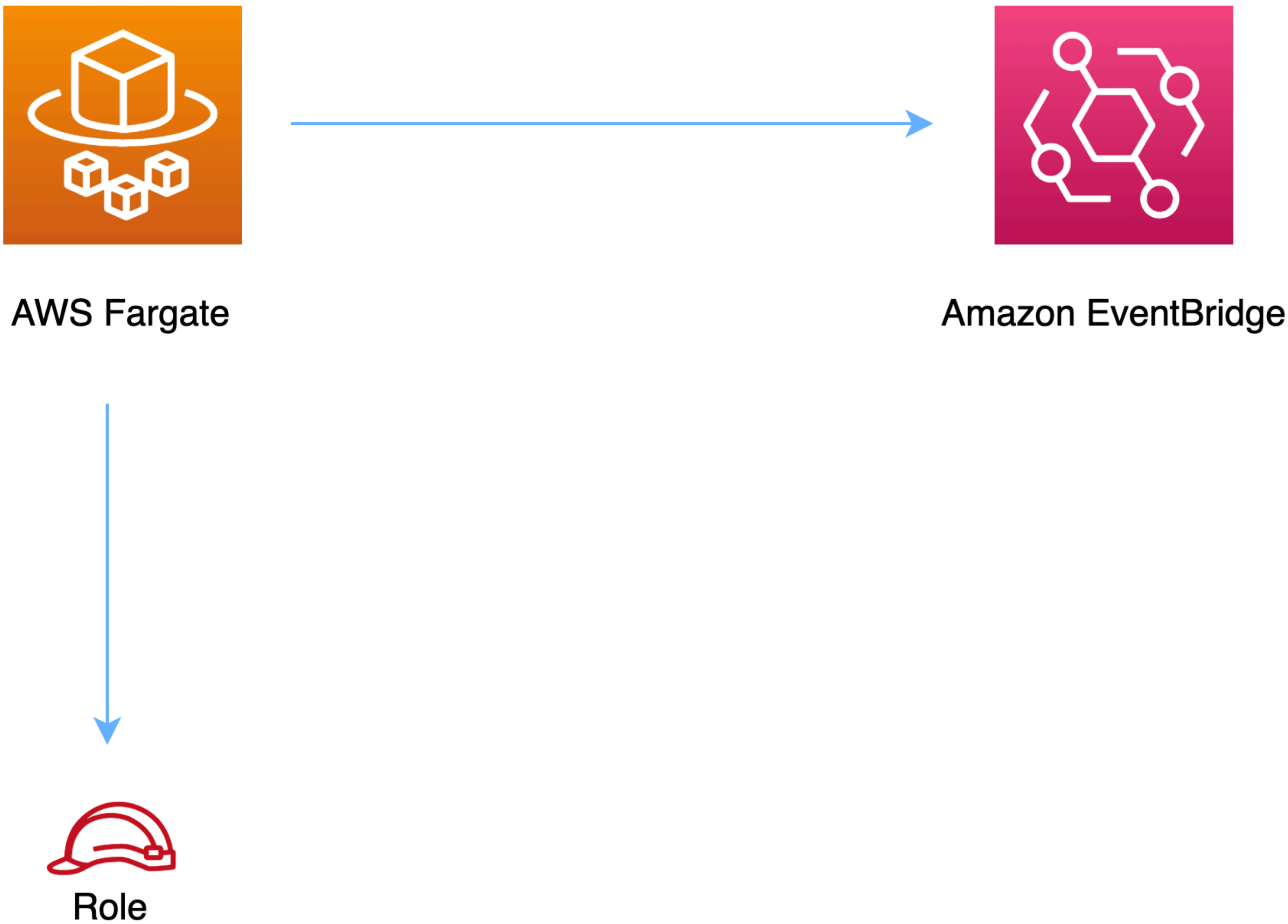
- Sets up an AWS Fargate service
  - Uses the existing service if provided
  - Creates a new service if none provided.
    - Service will run in isolated subnets if available, then private subnets if available and finally public subnets
- Adds environment variables to the container with the Name of the event bus
  - Default: EVENTBUS_NAME
- Add permissions to the container IAM role allowing it to put events in the EventBridge event bus

Amazon EventBridge Event Bus

- Sets up an Amazon EventBridge event bus

  - Uses an existing event bus if one is provided, otherwise creates a new one if `eventBusProps` is provided
  - If neither `eventBusProps` nor `existingEventBusInterface` is provided, the construct will use the `default` event bus.
- Adds an Interface Endpoint to the VPC for EventBridge (the service by default runs in Isolated or Private subnets)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-fargate-eventbridge](#)

aws-fargate-kinesisfirehose

STABILITY EXPERIMENTAL

Language	Package
Python	`aws_solutions_constructs.aws_fargate_kinesisfirehose`
Typescript	`@aws-solutions-constructs/aws-fargate-kinesisfirehose`
Java	`software.amazon.awsconstructs.services.fargatekinesisfirehose`

Overview

This AWS Solutions Construct deploys an AWS Fargate Service that can put records on an Amazon Firehose Delivery Stream.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { FargateToKinesisFirehoseProps } from '@aws-solutions-constructs/aws-
fargate-kinesisfirehose';
import * as fargate from 'aws-cdk-lib/aws-fargate';
```

```
// The construct requires an existing Firehose Delivery Stream, this can be created
 in raw CDK or extracted
// from a previously instantiated construct that created an Firehose Delivery Stream
const existingFirehoseDeliveryStream =
 previouslyCreatedKinesisFirehoseToS3Construct.kinesisFirehose;

new FargateToKinesisFirehose(this, 'FargateToKinesisFirehose', {
  publicApi: true,
  ecrRepositoryArn: "arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-repo",
  existingKinesisFirehose: existingFirehoseDeliveryStream
});
```

Python

```
from aws_solutions_constructs.aws_fargate_kinesisfirehose import
 FargateToKinesisFirehose
from aws_cdk import (
    aws_fargate as _fargate,
    Stack
)
from constructs import Construct

# The construct requires an existing Firehose Delivery Stream, this can be created
 in raw CDK or extracted
# from a previously instantiated construct that created an Firehose Delivery Stream
existingFirehoseDeliveryStream =
 previouslyCreatedKinesisFirehoseToS3Construct.kinesisFirehose;

FargateToKinesisFirehose(self, 'FargateToKinesisFirehose',
                          public_api=True,
                          ecr_repository_arn="arn:aws:ecr:us-
east-1:123456789012:repository/your-ecr-repo",
                          existingKinesisFirehose=existingFirehoseDeliveryStream
                       )
```

Java

```
import software.constructs.Construct;
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.fargate.*;
```

```
import software.amazon.awscdk.services.fargate.eventsources.*;
import software.amazon.awscdk.services.fargate.Runtime;
import software.amazon.awsconstructs.services.fargatekinesisfirehose.*;

// The construct requires an existing Firehose Delivery Stream, this can be created
 in raw CDK or extracted
// from a previously instantiated construct that created an Firehose Delivery Stream
existingFirehoseDeliveryStream =
 previouslyCreatedKinesisFirehoseToS3Construct.kinesisFirehose;

new FargateToKinesisFirehose(this, "FargateToKinesisFirehose", new
 FargateToKinesisFirehoseProps.Builder()
        .publicApi(true)
        .ecrRepositoryArn("arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-
repo")
        .existingKinesisFirehose(existingFirehoseDeliveryStream)
        .build());
```

Pattern Construct Props

Name	Type	Description
publicApi	`boolean`	True if the VPC provisioned by this construct should contain Public/Private Subnets, otherwise False for the VPC to contain Isolated Subnets only. Note this property is ignored if an existing VPC is specified in the existingVpc property. If you are getting a container from a public repo, this must be true so the repo can be accessed from the network.
vpcProps?	[`ec2.VpcProps`]()	Optional custom properties for a new VPC the construct will create. Providing both this and `existingVpc` is

Name	Type	Description
		an error. An Amazon Kinesis Firehose Interface Endpoint will be added to this VPC.
existingVpc?	[`ec2.IVpc`]()	An existing VPC in which to deploy the Fargate Service. Providing both this and `vpcProps` is an error. If the client provides an existing Fargate Service in the `existingFargateServiceObject` property, this value must be the VPC where the service is running. An Amazon Kinesis Firehose Interface Endpoint will be added to this VPC.
clusterProps?	[`ecs.ClusterProps`]()	Optional properties to create a new ECS cluster. To provide an existing cluster, use the cluster attribute of fargateServiceProps.
ecrRepositoryArn?	`string`	The arn of an ECR Repository containing the image to use to generate the containers. Either this or the image property of containerDefinitionProps must be provided. format: arn:aws:ecr:*region*:*account number*:repository/*Repository Name*

Name	Type	Description
ecrImageVersion?	`string`	The version of the image to use from the repository. Defaults to "Latest"
containerDefinitionProps?	`ecs.ContainerDefinitionProps | any`	Optional props to define the container created for the Fargate Service. (defaults found in fargate-defaults.ts)
fargateTaskDefinitionProps?	`ecs.FargateTaskDefinitionProps | any`	Optional props to define the Fargate Task Definition for this construct. (defaults found in fargate-defaults.ts)
fargateServiceProps?	`ecs.FargateServiceProps | any`	Optional values to override default Fargate Task definition properties (fargate-defaults.ts). The construct will default to launching the service is the most isolated subnets available (precedence: Isolated, Private and Public). Override those and other defaults here.
existingFargateServiceObject?	`ecs.FargateService`	A Fargate Service already instantiated (probably by another Solutions Construct). If this is specified, then no props defining a new service can be provided, including: ecrImageVersion, containerDefinitionProps, fargateTaskDefinitionProps, ecrRepositoryArn, fargateServiceProps, clusterProps

Name	Type	Description
existingContainerDefinitionObject?	[ecs.ContainerDefinition]()	A container definition already instantiated as part of a Fargate service. This must be the container in the `existingFargateServiceObject`.
existingKinesisFirehose	[kinesisfirehose.CfnDeliveryStream]()	An existing Kinesis Firehose Delivery Stream to which the Fargate container can put data. Note - the delivery stream construct must have already been created and have the deliveryStreamName set. This construct will *not* create a new Delivery Stream.
firehoseEnvironmentVariableName?	`string`	Optional Name for the Fargate container environment variable set to the name of the delivery stream. Default: FIREHOSE_DELIVERYSTREAM_NAME

Pattern Properties

Name	Type	Description
vpc	[ec2.IVpc]()	The new or existing VPC used by the construct.
fargateService	[ecs.FargateService]()	The new or existing AWS Fargate service used by this construct.

Name	Type	Description
container	[ecs.ContainerDefinition](#)	The container associated with the AWS Fargate service in the service property.
kinesisFirehose	[kinesisfirehose.CfnDeliveryStream](#)	The Kinesis Firehose Delivery Stream used by the construct.

Default settings

Out of the box implementation of the Construct without any overrides will set the following defaults:

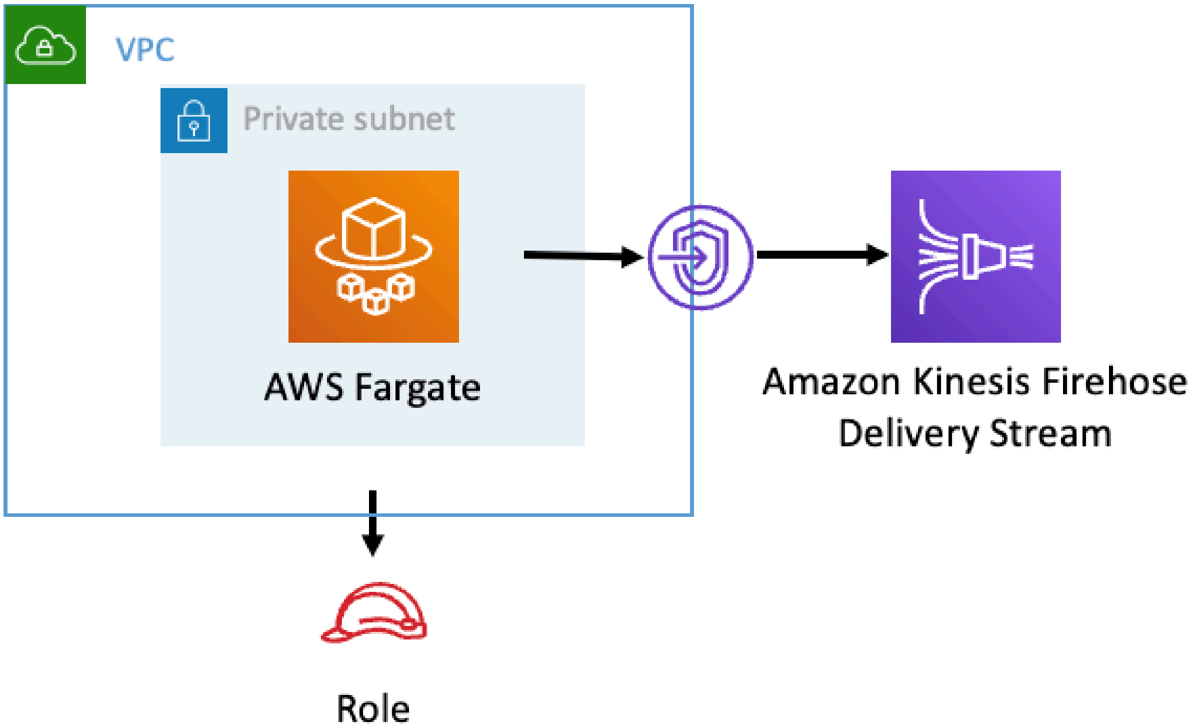
AWS Fargate Service

- An AWS Fargate Service running in the isolated subnets of a new VPC
- Minimally-permissive IAM role for the Fargate Service to put records on the Firehose Delivery Stream
- Sets an Environment Variable named FIREHOSE_DELIVERYSTREAM_NAME that holds the Firehose Delivery Stream Name, which is a required property of the Kinesis Firehose SDK when making calls to it

Amazon Firehose Delivery Stream

- This construct must be provided a configured Firehose Data Stream construct, it does not change this Stream.

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-fargate-kinesisfirehose

aws-fargate-kinesisstreams

STABILITY EXPERIMENTAL

Language	Package
Python	aws_solutions_constructs.aw s_fargate_kinesisstreams
Typescript	@aws-solutions-constructs/aws-fargate-kinesisstreams
Java	software.amazon.awsconstruc ts.services.fargatekinesisstreams

Overview

This AWS Solutions Construct deploys an AWS Fargate Service that can put records on an Amazon Kinesis Data Stream.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { FargateToKinesisStreamsProps } from '@aws-solutions-constructs/aws-fargate-
kinesisstreams';
import * as fargate from 'aws-cdk-lib/aws-fargate';

new FargateToKinesisStreams(this, 'FargateToKinesisStreams', {
  publicApi: true,
  ecrRepositoryArn: "arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-repo",
});
```

Python

```
from aws_solutions_constructs.aws_fargate_kinesisstreams import
 FargateToKinesisStreams
from aws_cdk import (
    aws_fargate as _fargate,
```

```
    Stack
)
from constructs import Construct

FargateToKinesisStreams(self, 'FargateToKinesisStreams',
                          public_api=True,
                          ecr_repository_arn="arn:aws:ecr:us-
east-1:123456789012:repository/your-ecr-repo"
                       )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.fargate.*;
import software.amazon.awscdk.services.fargate.eventsources.*;
import software.amazon.awscdk.services.fargate.Runtime;
import software.amazon.awsconstructs.services.fargatekinesisstreams.*;

new FargateToKinesisStreams(this, "FargateToKinesisStreams", new
 FargateToKinesisStreamsProps.Builder()
        .publicApi(true)
        .ecrRepositoryArn("arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-
repo")
        .build());
```

Pattern Construct Props

Name	Type	Description
publicApi	`boolean`	True if the VPC provisioned by this construct should contain Public/Private Subnets, otherwise False for the VPC to contain Isolated Subnets only. Note this property is ignored if

Name	Type	Description
		an existing VPC is specified in the `existingVpc` property.
vpcProps?	[`ec2.VpcProps`]()	Optional custom properties for a new VPC the construct will create. Providing both this and `existingVpc` is an error. An Amazon Kinesis Streams Interface Endpoint will be added to this VPC.
existingVpc?	[`ec2.IVpc`]()	An existing VPC in which to deploy the Fargate Service. Providing both this and `vpcProps` is an error. If the client provides an existing Fargate Service in the `existingFargateServiceObject` property, this value must be the VPC where the service is running. An Amazon Kinesis Streams Interface Endpoint will be added to this VPC.
clusterProps?	[`ecs.ClusterProps`]()	Optional properties to create a new ECS cluster. To provide an existing cluster, use the cluster attribute of fargateServiceProps.

Name	Type	Description
ecrRepositoryArn?	`string`	The arn of an ECR Repository containing the image to use to generate the containers. Either this or the image property of containerDefinitionProps must be provided. format: arn:aws:ecr:*region*:*account number*:repository/*Repository Name*
ecrImageVersion?	`string`	The version of the image to use from the repository. Defaults to "Latest"
containerDefinitionProps?	[`ecs.ContainerDefinitionProps | any`]()	Optional props to define the container created for the Fargate Service. (defaults found in fargate-defaults.ts)
fargateTaskDefinitionProps?	[`ecs.FargateTaskDefinitionProps | any`]()	Optional props to define the Fargate Task Definition for this construct. (defaults found in fargate-defaults.ts)
fargateServiceProps?	[`ecs.FargateServiceProps | any`]()	Optional values to override default Fargate Task definition properties (fargate-defaults.ts). The construct will default to launching the service is the most isolated subnets available (precedence: Isolated, Private and Public). Override those and other defaults here.

Name	Type	Description
existingFargateServiceObject?	`ecs.FargateService`	A Fargate Service already instantiated (probably by another Solutions Construct). If this is specified, then no props defining a new service can be provided, including: ecrImageVersion, containerDefinitionProps, fargateTaskDefinitionProps, ecrRepositoryArn, fargateServiceProps, clusterProps
existingContainerDefinitionObject?	`ecs.ContainerDefinition`	A container definition already instantiated as part of a Fargate service. This must be the container in the `existingFargateServiceObject`.
existingStreamObj?	`kinesis.Stream`	Existing instance of a Kinesis Data Stream. Providing both this and `kinesisStreamProps` will cause an error.
kinesisStreamProps?	`kinesis.StreamProps`	Optional user-provided props to override the default props for the Kinesis Data Stream. Providing both this and `existingStreamObj` will cause an error.
createCloudWatchAlarms	`boolean`	Whether to create recommended CloudWatch Alarms for the Kinesis Stream (defaults to true).

Name	Type	Description
streamEnvironmentVariableName?	`string`	Optional Name to override the Fargate Service default environment variable name that holds the Kinesis Data Stream name value. Default: KINESIS_DATASTREAM_NAME

Pattern Properties

Name	Type	Description
vpc	`ec2.IVpc`	The new or existing VPC used by the construct.
service	`ecs.FargateService`	The new or existing AWS Fargate service used by this construct.
container	`ecs.ContainerDefinition`	The container associated with the AWS Fargate service in the service property.
kinesisStream	`kinesis.Stream`	The new or existing Kinesis Data Stream used by this construct.
cloudwatchAlarms?	`cloudwatch.Alarm[]`	Returns the CloudWatch Alarms created to monitor the Kinesis Data Stream.

Default settings

Out of the box implementation of the Construct without any overrides will set the following defaults:

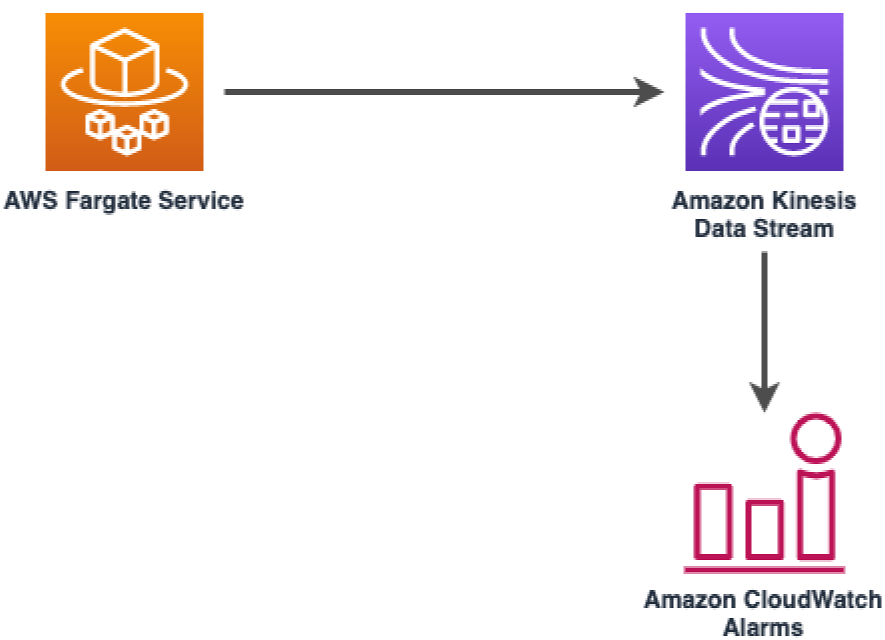
AWS Fargate Service

- An AWS Fargate Service running in the isolated subnets of a new VPC
- Minimally-permissive IAM role for the Fargate Service to put records on the Kinesis Data Stream
- Sets an Environment Variable named KINESIS_DATASTREAM_NAME that holds the Kinesis Data Stream Name, which is a required property of the Kinesis Data Streams SDK when making calls to it

Amazon Kinesis Stream

- Enable server-side encryption for the Kinesis Data Stream using an AWS Managed CMK
- Deploy best practices CloudWatch Alarms for the Kinesis Data Stream
- An Interface Endpoint on the VPC for private communication between the Fargate Service and the Kinesis Data Stream

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-fargate-kinesisstreams

aws-fargate-opensearch

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_fargate_opensearch`
Typescript	`@aws-solutions-constructs/aws-fargate-opensearch`
Java	`software.amazon.awsconstructs.services.fargateopensearch`

Overview

This AWS Solutions Construct implements an AWS Fargate service that can write/read to an Amazon OpenSearch Service domain.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { FargateToOpenSearch, FargateToOpenSearchProps } from '@aws-solutions-
constructs/aws-fargate-opensearch';

const constructProps: FargateToOpenSearchProps = {
  publicApi: true,
  ecrRepositoryArn: "arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-repo",
  openSearchDomainName: 'testdomain',
  // TODO: Ensure the Cognito domain name is globally unique
  cognitoDomainName: 'globallyuniquedomain' + Aws.ACCOUNT_ID
};

new FargateToOpenSearch(this, 'test-construct', constructProps);
```

Python

```
from aws_solutions_constructs.aws_fargate_opensearch import FargateToOpenSearch,
 FargateToOpenSearchProps
from aws_cdk import (
    Stack
)
from constructs import Construct

FargateToOpenSearch(self, 'test_construct',
            public_api=True,
            ecr_repository_arn="arn:aws:ecr:us-east-1:123456789012:repository/your-
ecr-repo",
            open_search_domain_name='testdomain',
            # TODO: Ensure the Cognito domain name is globally unique
            cognito_domain_name='globallyuniquedomain' + Aws.ACCOUNT_ID)
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.fargateopensearch.*;
```

```
new FargateToOpenSearch(this, "test_construct", new
 FargateToOpenSearchProps.Builder()
        .publicApi(true)
        .ecrRepositoryArn("arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-
repo"
        .openSearchDomainName("testdomain")
        // TODO: Ensure the Cognito domain name is globally unique
        .cognitoDomainName("globallyuniquedomain" + Aws.ACCOUNT_ID)
        .build());
```

Pattern Construct Props

Name	Type	Description
publicApi	`boolean`	Whether the construct is deploying a private or public API. This has implications for the VPC.
vpcProps?	[`ec2.VpcProps`]()	Optional custom properties for a VPC the construct will create. This VPC will be used by any Private Hosted Zone the construct creates (that's why loadBalancerProps and privateHostedZoneProps can't include a VPC). Providing both this and existingVpc is an error.
existingVpc?	[`ec2.IVpc`]()	An existing VPC in which to deploy the construct. Providing both this and vpcProps is an error. If the client provides an existing load balancer and/or existing Private Hosted Zone, those constructs must exist in this VPC.

Name	Type	Description
clusterProps?	`ecs.ClusterProps`	Optional properties to create a new ECS cluster. To provide an existing cluster, use the cluster attribute of fargateServiceProps.
ecrRepositoryArn?	`string`	The arn of an ECR Repository containing the image to use to generate the containers. Either this or the image property of containerDefinitionProps must be provided. format: arn:aws:ecr:*region*:*account number*:repository/*Repository Name*
ecrImageVersion?	`string`	The version of the image to use from the repository. Defaults to "Latest".
containerDefinitionProps?	`ecs.ContainerDefinitionProps | any`	Optional props to define the container created for the Fargate Service (defaults found in fargate-defaults.ts).
fargateTaskDefinitionProps?	`ecs.FargateTaskDefinitionProps | any`	Optional props to define the Fargate Task Definition for this construct (defaults found in fargate-defaults.ts).

Name	Type	Description
fargateServiceProps?	[ecs.FargateService Props | any](#)	Optional values to override default Fargate Task definition properties (fargate-defaults.ts). The construct will default to launching the service is the most isolated subnets available (precedence: Isolated, Private and Public). Override those and other defaults here.
existingFargateServiceObject?	[ecs.FargateService](#)	A Fargate Service already instantiated (probably by another Solutions Construct). If this is specified, then no props defining a new service can be provided, including: ecrImageVersion, containerDefinitionProps, fargateTaskDefinitionProps, ecrRepositoryArn, fargateServiceProps, clusterProps.
existingContainerDefinitionObject?	[ecs.ContainerDefinition](#)	A container definition already instantiated as part of a Fargate service. This must be the container in the existingFargateServiceObject.
openSearchDomainProps?	[opensearchservice.CfnDomainProps](#)	Optional user provided props to override the default props for the OpenSearch Service.
openSearchDomainName	`string`	Domain name for the OpenSearch Service.

Name	Type	Description
cognitoDomainName?	`string`	Optional Amazon Cognito domain name. If omitted the Amazon Cognito domain will default to the OpenSearch Service domain name.
createCloudWatchAlarms?	`boolean`	Whether to create the recommended CloudWatch alarms.
domainEndpointEnvironmentVariableName?	`string`	Optional name for the OpenSearch Service domain endpoint environment variable set for the Lambda function. Default is `DOMAIN_ENDPOINT` .

Pattern Properties

Name	Type	Description
vpc	`ec2.IVpc`	The VPC used by the construct (whether created by the construct or provided by the client).
service	`ecs.FargateService`	The AWS Fargate service used by this construct (whether created by this construct or passed to this construct at initialization).
container	`ecs.ContainerDefinition`	The container associated with the AWS Fargate service in the service property.

Name	Type	Description
userPool	`cognito.UserPool`	Returns an instance of `cognito.UserPool` created by the construct.
userPoolClient	`cognito.UserPoolClient`	Returns an instance of `cognito.UserPoolClient` created by the construct.
identityPool	`cognito.CfnIdentityPool`	Returns an instance of `cognito.CfnIdentityPool` created by the construct.
openSearchDomain	`opensearchservice.CfnDomain`	Returns an instance of `opensearch.CfnDomain` created by the construct.
openSearchRole	`iam.Role`	Returns an instance of `iam.Role` created by the construct for `opensearch.CfnDomain`.
cloudWatchAlarms?	`cloudwatch.Alarm[]`	Returns a list of `cloudwatch.Alarm` created by the construct.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS Fargate Service

- Sets up an AWS Fargate service
  - Uses the existing service if provided
  - Creates a new service if none provided

    - Service will run in isolated subnets if available, then private subnets if available and finally public subnets
- Adds environment variables to the container with the OpenSearch Service domain endpoint
- Add permissions to the container IAM role allowing it to write/read to the OpenSearch Service domain endpoint

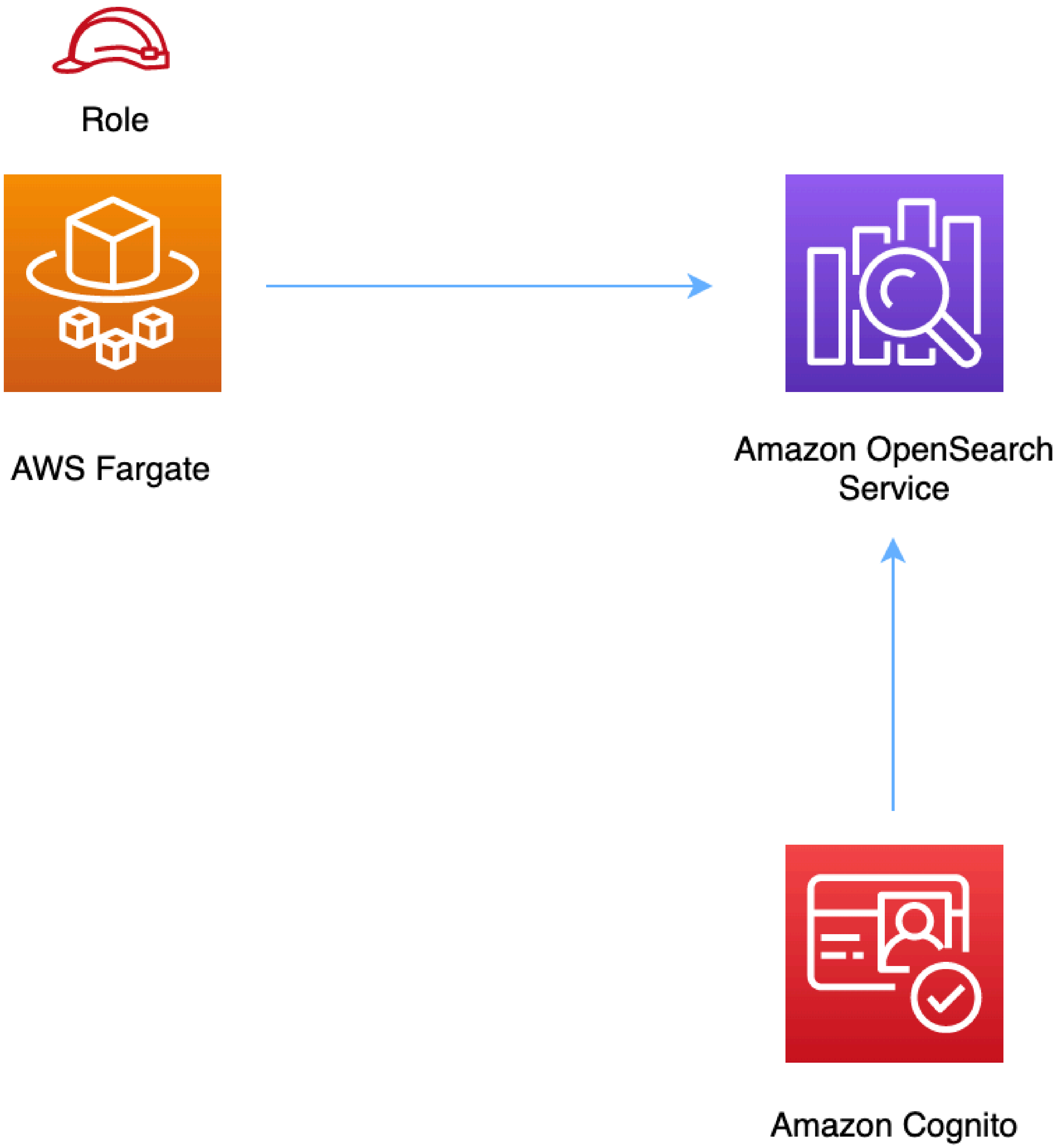
Amazon Cognito

- Set password policy for User Pools
- Enforce the advanced security mode for User Pools

Amazon OpenSearch Service

- Deploy best practices CloudWatch Alarms for the OpenSearch Service domain
- Secure the OpenSearch Service dashboard access with Cognito User Pools
- Enable server-side encryption for OpenSearch Service domain using AWS managed KMS Key
- Enable node-to-node encryption for the OpenSearch Service domain
- Configure the cluster for the OpenSearch Service domain

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-fargate-opensearch](#)

aws-fargate-s3

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_fargate_s3`
Typescript	`@aws-solutions-constructs/aws-fargate-s3`
Java	`software.amazon.awsconstructs.services.fargates3`

Overview

This AWS Solutions Construct implements an AWS Fargate service that can write/read to an Amazon S3 Bucket

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { FargateToS3, FargateToS3Props } from '@aws-solutions-constructs/aws-
fargate-s3';

const constructProps: FargateToS3Props = {
  publicApi: true,
  ecrRepositoryArn: "arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-repo",
};

new FargateToS3(this, 'test-construct', constructProps);
```

Python

```
from aws_solutions_constructs.aws_fargate_s3 import FargateToS3, FargateToS3Props
from aws_cdk import (
    Stack
)
from constructs import Construct

FargateToS3(self, 'test_construct',
            public_api=True,
            ecr_repository_arn="arn:aws:ecr:us-east-1:123456789012:repository/your-
ecr-repo")
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.fargates3.*;

new FargateToS3(this, "test_construct", new FargateToS3Props.Builder()
        .publicApi(true)
        .ecrRepositoryArn("arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-
repo")
        .build());
```

Pattern Construct Props

Name	Type	Description
publicApi	boolean	Whether the construct is deploying a private or public API. This has implications for the VPC.
vpcProps?	ec2.VpcProps	Optional custom properties for a VPC the construct will create. This VPC will be used by any Private Hosted Zone the construct creates (that's why loadBalancerProps and privateHostedZoneProps can't include a VPC). Providing both this and existingVpc is an error.
existingVpc?	ec2.IVpc	An existing VPC in which to deploy the construct. Providing both this and vpcProps is an error. If the client provides an existing load balancer and/or existing Private Hosted Zone, those constructs must exist in this VPC.
clusterProps?	ecs.ClusterProps	Optional properties to create a new ECS cluster. To provide an existing cluster, use the cluster attribute of fargateServiceProp s.
ecrRepositoryArn?	string	The arn of an ECR Repository containing the image to use to generate the containers. Either this or the image property

Name	Type	Description
		of containerDefinitionProps must be provided. format: arn:aws:ecr:*region*:*account number*:repository/*Repository Name*
ecrImageVersion?	string	The version of the image to use from the repository. Defaults to "Latest"
containerDefinitionProps?	[ecs.ContainerDefinitionProps | any]()	Optional props to define the container created for the Fargate Service (defaults found in fargate-defaults.ts)
fargateTaskDefinitionProps?	[ecs.FargateTaskDefinitionProps | any]()	Optional props to define the Fargate Task Definition for this construct (defaults found in fargate-defaults.ts)
fargateServiceProps?	[ecs.FargateServiceProps | any]()	Optional values to override default Fargate Task definition properties (fargate-defaults.ts). The construct will default to launching the service is the most isolated subnets available (precedence: Isolated, Private and Public). Override those and other defaults here.

Name	Type	Description
existingFargateSer viceObject?	ecs.FargateService	A Fargate Service already instantiated (probably by another Solutions Construct). If this is specified, then no props defining a new service can be provided, including: ecrImageVersion, container DefinitionProps, fargateTa skDefinitionProps, ecrReposi toryArn, fargateServiceProps, clusterProps
existingContainerD efinitionObject?	ecs.ContainerDefinition	A container definition already instantiated as part of a Fargate service. This must be the container in the existingF argateServiceObject
existingBucketInterface?	`s3.IBucket`	Existing S3 Bucket interface . Providing this property and `bucketProps` results in an error.
bucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the S3 Bucket.
loggingBucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the S3 Logging Bucket.

Name	Type	Description
logS3AccessLogs?	boolean	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with associated storage costs for the logs. Enabling Access Logging is a best practice. default - true
bucketPermissions?	`string[]`	Optional bucket permissions to grant to the Fargate service. One or more of the following may be specified: `Delete`, `Read`, and `Write`. Default is ["Read", "Write"] which includes `[s3:GetObject*, s3:GetBucket*, s3:List*, s3:DeleteObject*, s3:PutObject*, s3:Abort*]` .
bucketArnEnvironmentVariableName?	string	Optional Name for the container environment variable set to the bucket ARN. Default: S3_BUCKET_ARN
bucketEnvironmentVariableName?	string	Optional Optional Name for the container environment variable set to the bucket name. Default: S3_BUCKET_NAME

Pattern Properties

Name	Type	Description
vpc	ec2.IVpc	The VPC used by the construct (whether created by the construct or provided by the client)
service	ecs.FargateService	The AWS Fargate service used by this construct (whether created by this construct or passed to this construct at initialization)
container	ecs.ContainerDefinition	The container associated with the AWS Fargate service in the service property.
s3Bucket?	s3.IBucket	Returns an instance of s3.Bucket created by the construct
s3BucketInterface	`s3.IBucket`	Returns an instance of s3.IBucke t created by the construct
s3LoggingBucket?	s3.Bucket	Returns an instance of s3.Bucket created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS Fargate Service

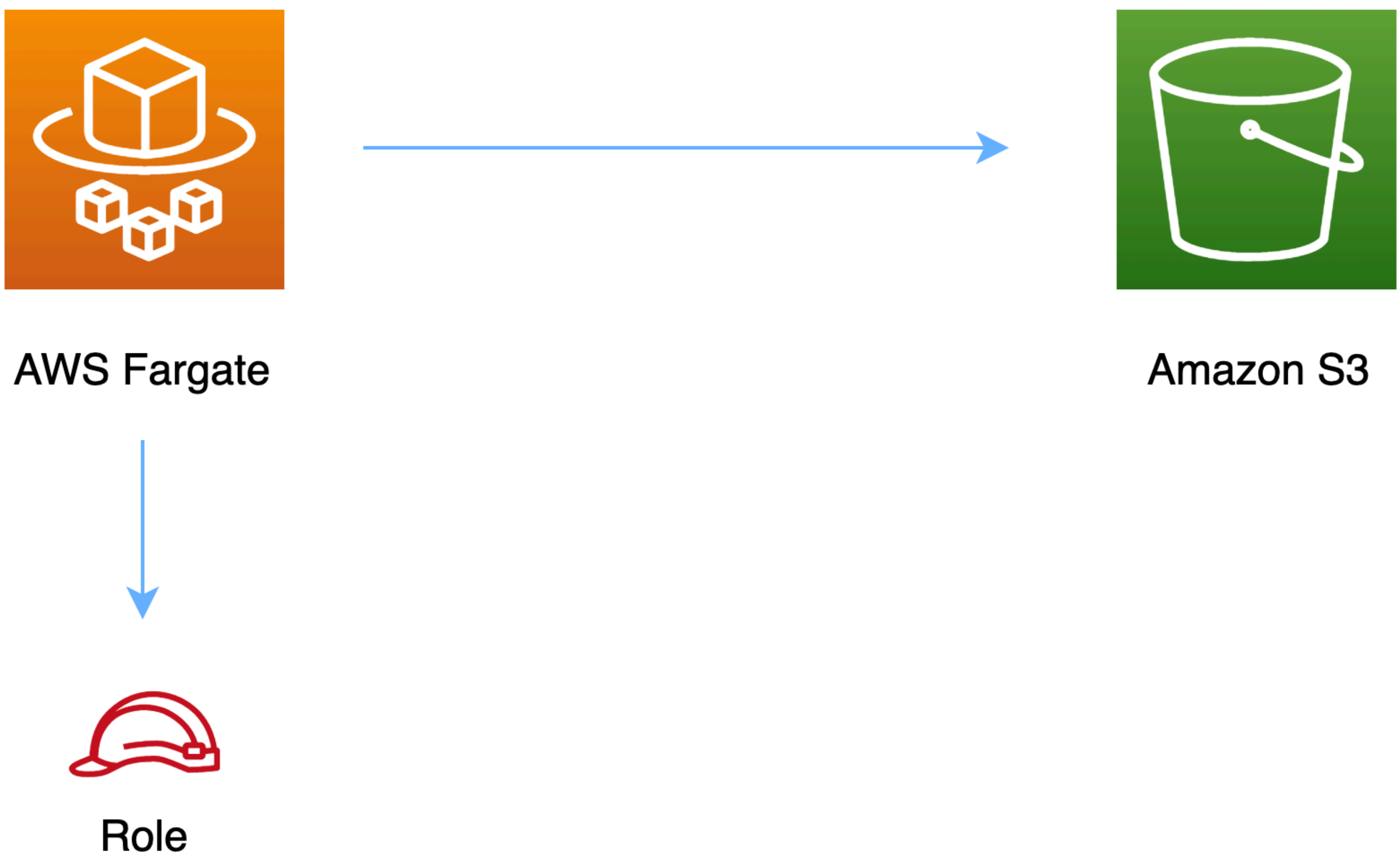
- Sets up an AWS Fargate service

- Uses the existing service if provided
- Creates a new service if none provided.
  - Service will run in isolated subnets if available, then private subnets if available and finally public subnets
- Adds environment variables to the container with the ARN and Name of the S3 Bucket
- Add permissions to the container IAM role allowing it to publish to the S3 Bucket

Amazon S3 Bucket

- Sets up an Amazon S3 Bucket
  - Uses an existing bucket if one is provided, otherwise creates a new one
- Adds an Interface Endpoint to the VPC for S3 (the service by default runs in Isolated or Private subnets)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-fargate-s3

aws-fargate-secretsmanager

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_fargate_secretsmanager`
Typescript	`@aws-solutions-constructs/aws-fargate-secretsmanager`
Java	`software.amazon.awsconstructs.services.fargatesecretsmanager`

Overview

This AWS Solutions Construct implements an AWS Fargate service that can write/read to an AWS Secrets Manager

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { FargateToSecretsmanager, FargateToSecretsmanagerProps } from '@aws-
solutions-constructs/aws-fargate-secretsmanager';

const constructProps: FargateToSecretsmanagerProps = {
  publicApi: true,
  ecrRepositoryArn: "arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-repo",
};

new FargateToSecretsmanager(stack, 'test-construct', constructProps);
```

Python

```
from aws_solutions_constructs.aws_fargate_secretsmanager import
 FargateToSecretsmanager, FargateToSecretsmanagerProps
from aws_cdk import (
    Stack
)
from constructs import Construct

FargateToSecretsmanager(self, 'test_construct',
            public_api=True,
            ecr_repository_arn="arn:aws:ecr:us-east-1:123456789012:repository/your-
ecr-repo")
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.fargatesecretsmanager.*;

new FargateToSecretsmanager(this, "test-construct", new
 FargateToSecretsmanagerProps.Builder()
        .publicApi(true)
        .ecrRepositoryArn("arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-
repo")
        .build());
```

Pattern Construct Props

Name	Type	Description
publicApi	`boolean`	Whether the construct is deploying a private or public API. This has implications for the VPC.
vpcProps?	[`ec2.VpcProps`]()	Optional custom properties for a VPC the construct will create. This VPC will be used by any Private Hosted Zone the construct creates (that's why loadBalancerProps and privateHostedZoneProps can't include a VPC). Providing both this and existingVpc is an error.
existingVpc?	[`ec2.IVpc`]()	An existing VPC in which to deploy the construct. Providing both this and vpcProps is an error. If the client provides an existing load balancer and/or existing Private Hosted Zone, those constructs must exist in this VPC.
clusterProps?	[`ecs.ClusterProps`]()	Optional properties to create a new ECS cluster. To provide an existing cluster, use the cluster attribute of fargateServiceProp s.
ecrRepositoryArn?	`string`	The arn of an ECR Repository containing the image to use to generate the containers. Either this or the image property

Name	Type	Description
		of containerDefinitionProps must be provided. format: arn:aws:ecr:*region*:*account number*:repository/*Repository Name*
ecrImageVersion?	`string`	The version of the image to use from the repository. Defaults to "Latest"
containerDefinitionProps?	`ecs.ContainerDefinitionProps | any`	Optional props to define the container created for the Fargate Service (defaults found in fargate-defaults.ts)
fargateTaskDefinitionProps?	`ecs.FargateTaskDefinitionProps | any`	Optional props to define the Fargate Task Definition for this construct (defaults found in fargate-defaults.ts)
fargateServiceProps?	`ecs.FargateServiceProps | any`	Optional values to override default Fargate Task definition properties (fargate-defaults.ts). The construct will default to launching the service is the most isolated subnets available (precedence: Isolated, Private and Public). Override those and other defaults here.

Name	Type	Description
existingFargateSer viceObject?	ecs.FargateService	A Fargate Service already instantiated (probably by another Solutions Construct). If this is specified, then no props defining a new service can be provided, including: ecrImageVersion, container DefinitionProps, fargateTa skDefinitionProps, ecrReposi toryArn, fargateServiceProps, clusterProps
existingContainerD efinitionObject?	ecs.ContainerDefin ition	A container definition already instantiated as part of a Fargate service. This must be the container in the existingF argateServiceObject
secretProps?	secretsmanager.Sec retProps	Optional user provided props to override the default props for Secrets Manager
existingSecretObj?	secretsmanager.Secret	Existing instance of Secrets Manager Secret object, If this is set then the secretProps is ignored
grantWriteAccess?	boolean	Optional write access to the Secret for the Fargate service (Read-Only by default)
secretEnvironmentV ariableName?	string	Optional Name for the container environment variable set to the ARN of the secret. Default: SECRET_ARN

Pattern Properties

Name	Type	Description
vpc	`ec2.IVpc`	The VPC used by the construct (whether created by the construct or provided by the client)
service	`ecs.FargateService`	The AWS Fargate service used by this construct (whether created by this construct or passed to this construct at initialization)
container	`ecs.ContainerDefinition`	The container associated with the AWS Fargate service in the service property.
secret	`secretsmanager.Secret`	Returns an instance of `secretsmanager.Secret` created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS Fargate Service

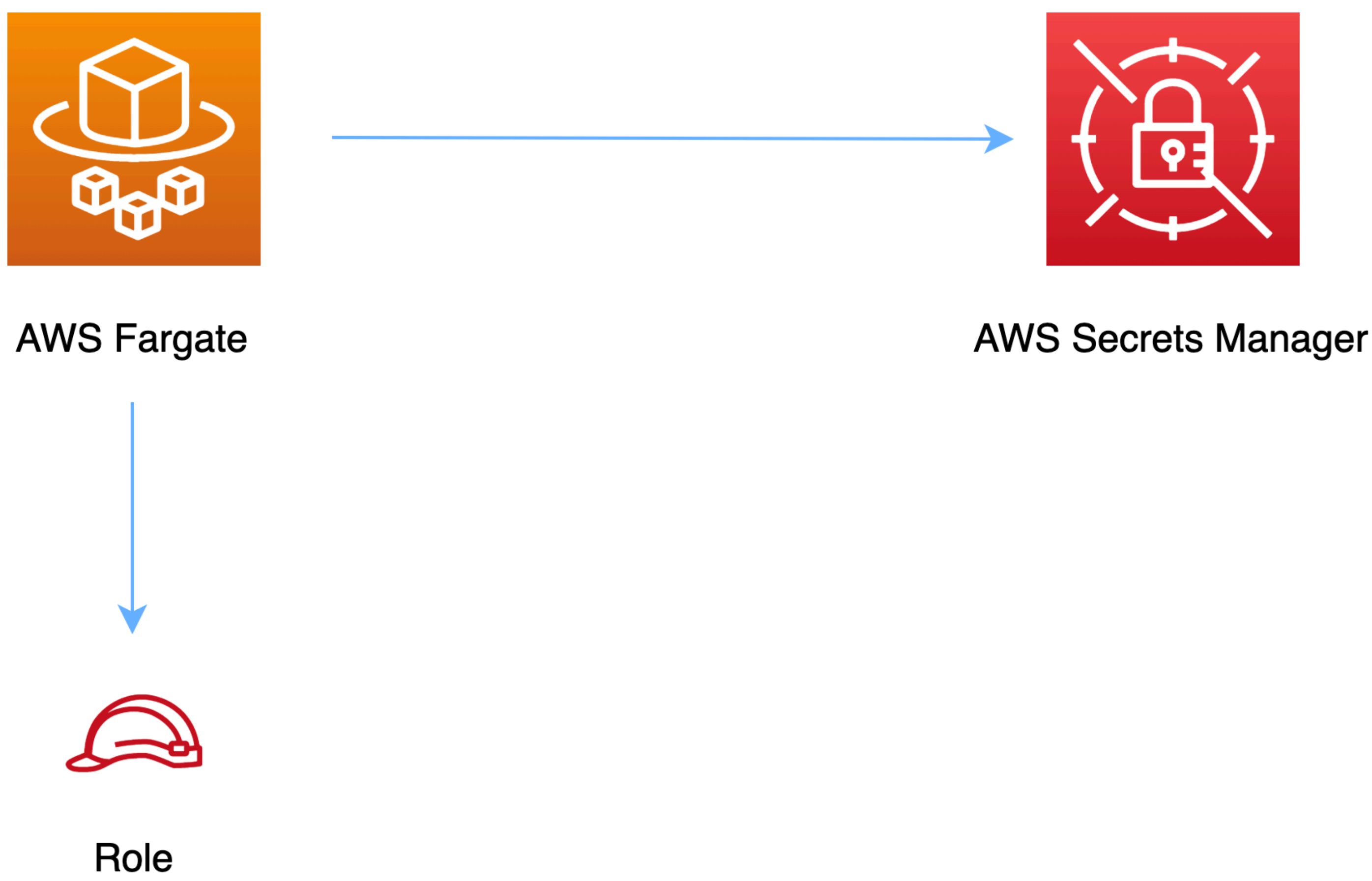
- Sets up an AWS Fargate service
  - Uses the existing service if provided
  - Creates a new service if none provided.
    - Service will run in isolated subnets if available, then private subnets if available and finally public subnets
  - Adds environment variables to the container with the ARN and Name of the Secrets Manager secret

- Add permissions to the container IAM role allowing it to publish to the Secrets Manager secret

Amazon Secrets Manager Secret

- Sets up an Amazon Secrets Manager secret
  - Uses an existing secret if one is provided, otherwise creates a new one
    - (default) random name
    - (default) random value
- Adds an Interface Endpoint to the VPC for Secrets Manager (the service by default runs in Isolated or Private subnets)
- Retain the Secret when deleting the CloudFormation stack

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-fargate-secretsmanager

aws-fargate-sns

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_fargate_sns`
Typescript	`@aws-solutions-constructs/aws-fargate-sns`
Java	`software.amazon.awsconstructs.services.fargatesns`

Overview

This AWS Solutions Construct implements an AWS Fargate service that can write to an Amazon SNS topic

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { FargateToSns, FargateToSnsProps } from '@aws-solutions-constructs/aws-
fargate-sns';

const constructProps: FargateToSnsProps = {
    publicApi: true,
    ecrRepositoryArn: "arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-repo"
};

new FargateToSns(this, 'test-construct', constructProps);
```

Python

```
from aws_solutions_constructs.aws_fargate_sns import FargateToSns, FargateToSnsProps
from aws_cdk import (
    Stack
)
from constructs import Construct

FargateToSns(self, 'test_construct',
            public_api=True,
            ecr_repository_arn="arn:aws:ecr:us-east-1:123456789012:repository/your-
ecr-repo")
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.fargatesns.*;

new FargateToSns(this, "test_construct", new FargateToSnsProps.Builder()
        .publicApi(true)
        .ecrRepositoryArn("arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-
repo")
        .build());
```

Pattern Construct Props

Name	Type	Description
publicApi	boolean	Whether the construct is deploying a private or public API. This has implications for the VPC.
vpcProps?	ec2.VpcProps	Optional custom properties for a VPC the construct will create. This VPC will be used by any Private Hosted Zone the construct creates (that's why loadBalancerProps and privateHostedZoneProps can't include a VPC). Providing both this and existingVpc is an error.
existingVpc?	ec2.IVpc	An existing VPC in which to deploy the construct. Providing both this and vpcProps is an error. If the client provides an existing load balancer and/or existing Private Hosted Zone, those constructs must exist in this VPC.
clusterProps?	ecs.ClusterProps	Optional properties to create a new ECS cluster. To provide an existing cluster, use the cluster attribute of fargateServiceProp s.
ecrRepositoryArn?	string	The arn of an ECR Repository containing the image to use to generate the containers. Either this or the image property

Name	Type	Description
		of containerDefinitionProps must be provided. format: arn:aws:ecr:*region*:*account number*:repository/*Repository Name*
ecrImageVersion?	string	The version of the image to use from the repository. Defaults to "Latest"
containerDefinitionProps?	[ecs.ContainerDefinitionProps | any]()	Optional props to define the container created for the Fargate Service (defaults found in fargate-defaults.ts)
fargateTaskDefinitionProps?	[ecs.FargateTaskDefinitionProps | any]()	Optional props to define the Fargate Task Definition for this construct (defaults found in fargate-defaults.ts)
fargateServiceProps?	[ecs.FargateServiceProps | any]()	Optional values to override default Fargate Task definition properties (fargate-defaults.ts). The construct will default to launching the service is the most isolated subnets available (precedence: Isolated, Private and Public). Override those and other defaults here.

Name	Type	Description
existingFargateSer viceObject?	ecs.FargateService	A Fargate Service already instantiated (probably by another Solutions Construct). If this is specified, then no props defining a new service can be provided, including: ecrImageVersion, container DefinitionProps, fargateTa skDefinitionProps, ecrReposi toryArn, fargateServiceProps, clusterProps
existingContainerD efinitionObject?	ecs.ContainerDefinition	A container definition already instantiated as part of a Fargate service. This must be the container in the existingF argateServiceObject
existingTopicObj?	sns.Topic	Existing instance of SNS Topic object, providing both this and `topicProps` will cause an error.
topicProps?	sns.TopicProps	Optional user provided properties to override the default properties for the SNS topic.
topicArnEnvironmen tVariableName?	string	Optional Name for the container environment variable set to the ARN of the topic. Default: SNS_TOPIC_ARN

Name	Type	Description
topicNameEnvironmentVariableName?	string	Optional Name for the container environment variable set to the name of the topic. Default: SNS_TOPIC_NAME
enableEncryptionWithCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the SNS Topic is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined: topicProps.masterKey, encryptionKey or encryptionKeyProps.
encryptionKey?	`kms.Key`	An optional, imported encryption key to encrypt the SNS Topic with.
encryptionKeyProps?	`kms.KeyProps`	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SNS Topic with.

Pattern Properties

Name	Type	Description
vpc	ec2.IVpc	The VPC used by the construct (whether created by the construct or provided by the client)
service	ecs.FargateService	The AWS Fargate service used by this construct (whether

Name	Type	Description
		created by this construct or passed to this construct at initialization)
container	ecs.ContainerDefinition	The container associated with the AWS Fargate service in the service property.
snsTopic	`sns.Topic`	Returns an instance of the SNS topic created by the pattern.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

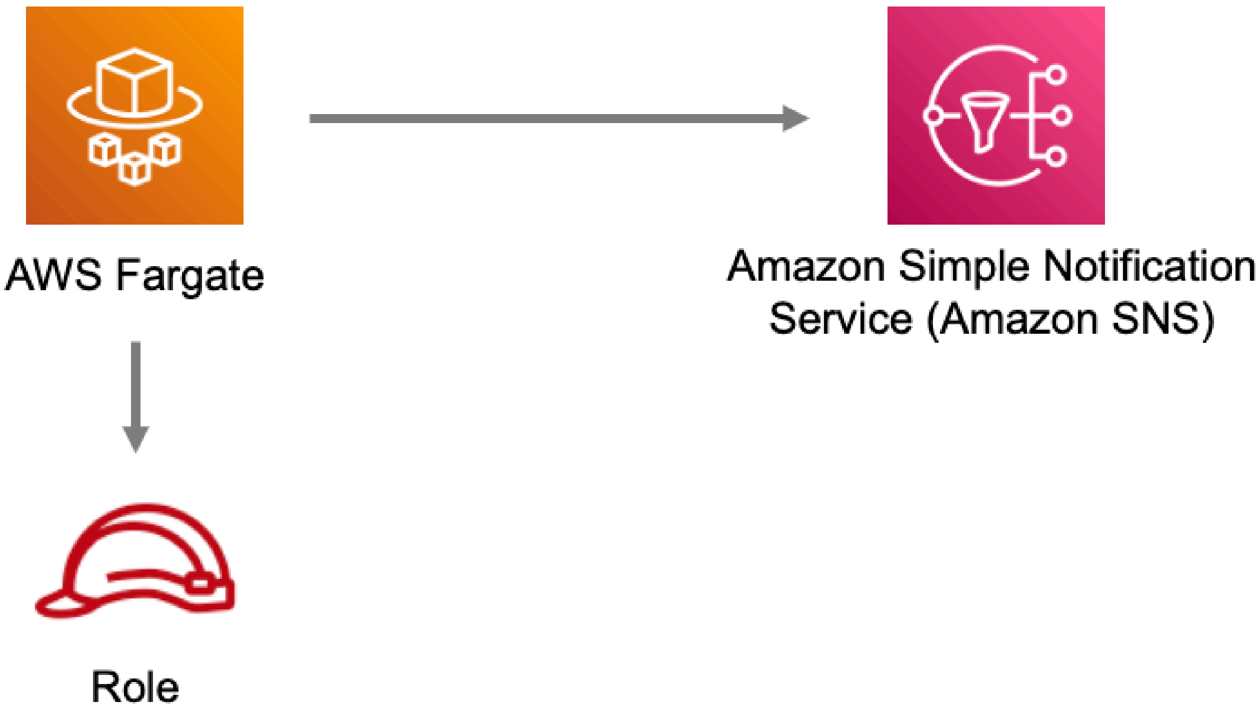
AWS Fargate Service

- Sets up an AWS Fargate service
  - Uses the existing service if provided
  - Creates a new service if none provided.
    - Service will run in isolated subnets if available, then private subnets if available and finally public subnets
  - Adds environment variables to the container with the ARN and Name of the SNS topic
  - Add permissions to the container IAM role allowing it to publish to the SNS topic

Amazon SNS Topic

- Sets up an Amazon SNS topic
  - Uses an existing topic if one is provided, otherwise creates a new one
- Adds an Interface Endpoint to the VPC for SNS (the service by default runs in Isolated or Private subnets)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-fargate-sns

aws-fargate-sqs

STABILITY EXPERIMENTAL

All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aw s_fargate_sqs
Typescript	@aws-solutions-constructs/aws-fargate-sqs
Java	software.amazon.awsconstruc ts.services.fargatesqs

Overview

This AWS Solutions Construct implements an AWS Fargate service that can write to an Amazon SQS queue

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { FargateToSqs, FargateToSqsProps } from '@aws-solutions-constructs/aws-
fargate-sqs';

const constructProps: FargateToSqsProps = {
    publicApi: true,
    ecrRepositoryArn: "arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-repo"
};

new FargateToSqs(this, 'test-construct', constructProps);
```

Python

```
from aws_solutions_constructs.aws_fargate_sqs import FargateToSqs, FargateToSqsProps
from aws_cdk import (
    Stack
```

```
)
from constructs import Construct

FargateToSqs(self, 'test_construct',
            public_api=True,
            ecr_repository_arn="arn:aws:ecr:us-east-1:123456789012:repository/your-
ecr-repo")
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.fargatesqs.*;

new FargateToSqs(this, "test_construct", new FargateToSqsProps.Builder()
        .publicApi(true)
        .ecrRepositoryArn("arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-
repo")
        .build());
```

Pattern Construct Props

Name	Type	Description
publicApi	boolean	Whether the construct is deploying a private or public API. This has implications for the VPC.
vpcProps?	[ec2.VpcProps]()	Optional custom properties for a VPC the construct will create. This VPC will be used by any Private Hosted Zone the construct creates (that's why loadBalancerProps and privateHostedZoneProps can't

Name	Type	Description
		include a VPC). Providing both this and existingVpc is an error.
existingVpc?	[ec2.IVpc]()	An existing VPC in which to deploy the construct. Providing both this and vpcProps is an error. If the client provides an existing load balancer and/or existing Private Hosted Zone, those constructs must exist in this VPC.
clusterProps?	[ecs.ClusterProps]()	Optional properties to create a new ECS cluster. To provide an existing cluster, use the cluster attribute of fargateServiceProp s.
ecrRepositoryArn?	string	The arn of an ECR Repository containing the image to use to generate the containers. Either this or the image property of containerDefinitionProps must be provided. format: arn:aws:ecr:*region*:*account number*:repository/*Repository Name*
ecrImageVersion?	string	The version of the image to use from the repository. Defaults to "Latest"
containerDefinitionProps?	[ecs.ContainerDefinitionProps | any]()	Optional props to define the container created for the Fargate Service (defaults found in fargate-defaults.ts)

Name	Type	Description
fargateTaskDefinitionProps?	ecs.FargateTaskDefinitionProps | any	Optional props to define the Fargate Task Definition for this construct (defaults found in fargate-defaults.ts)
fargateServiceProps?	ecs.FargateServiceProps | any	Optional values to override default Fargate Task definition properties (fargate-defaults.ts). The construct will default to launching the service is the most isolated subnets available (precedence: Isolated, Private and Public). Override those and other defaults here.
existingFargateServiceObject?	ecs.FargateService	A Fargate Service already instantiated (probably by another Solutions Construct). If this is specified, then no props defining a new service can be provided, including: ecrImageVersion, containerDefinitionProps, fargateTaskDefinitionProps, ecrRepositoryArn, fargateServiceProps, clusterProps
existingContainerDefinitionObject?	ecs.ContainerDefinition	A container definition already instantiated as part of a Fargate service. This must be the container in the existingFargateServiceObject

Name	Type	Description
existingQueueObj?	sqs.Queue	An optional, existing SQS queue to be used instead of the default queue. Providing both this and queueProps will cause an error.
queueProps?	sqs.QueueProps	Optional user-provided properties to override the default properties for the SQS queue.
deployDeadLetterQueue?	boolean	Whether to create a secondary queue to be used as a dead letter queue. Defaults to `true`.
deadLetterQueueProps?	sqs.QueueProps	Optional user-provided props to override the default props for the dead letter queue. Only used if the `deployDeadLetterQueue` property is set to true.
maxReceiveCount?	integer	The number of times a message can be unsuccessfully dequeued before being moved to the dead letter queue. Defaults to 15.
queueUrlEnvironmentVariableName?	string	Optional Name for the container environment variable set to the URL of the queue. Default: SQS_QUEUE_URL
queueArnEnvironmentVariableName?	string	Optional Name for the container environment variable set to the arn of the queue. Default: SQS_QUEUE_ARN

Name	Type	Description
queuePermissions?	`string[]`	Optional queue permissions to grant to the Fargate service. One or more of the following may be specified: `Read`,`Write`. Default is `Write`
enableEncryptionWithCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the queue is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined: queueProps.encryptionMasterKey, encryptionKey or encryptionKeyProps.
encryptionKey?	`kms.Key`	An optional, imported encryption key to encrypt the SQS Queue with.
encryptionKeyProps?	`kms.KeyProps`	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SQS queue with.

Pattern Properties

Name	Type	Description
vpc	ec2.IVpc	The VPC used by the construct (whether created by the construct or provided by the client)

Name	Type	Description
service	ecs.FargateService	The AWS Fargate service used by this construct (whether created by this construct or passed to this construct at initialization)
container	ecs.ContainerDefinition	The container associated with the AWS Fargate service in the service property.
sqsQueue	sqs.Queue	Returns an instance of the SQS queue created by the pattern.
deadLetterQueue?	sqs.Queue	Returns an instance of the dead letter queue created by the pattern, if one is deployed.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS Fargate Service

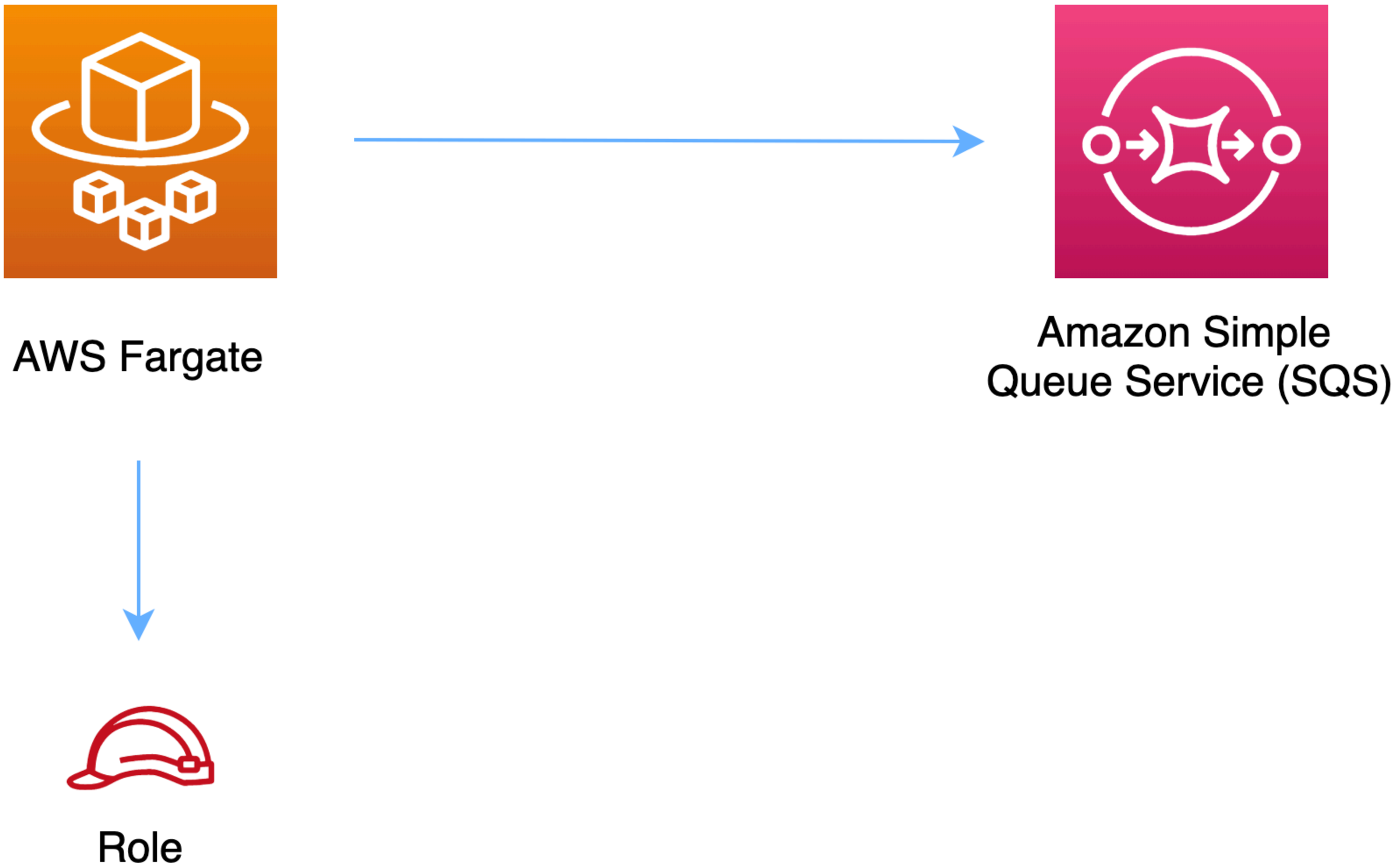
- Sets up an AWS Fargate service
  - Uses the existing service if provided
  - Creates a new service if none provided.
    - Service will run in isolated subnets if available, then private subnets if available and finally public subnets
  - Adds environment variables to the container with the name of the SQS queue
  - Add permissions to the container IAM role allowing it to publish to the SQS queue

Amazon SQS Queue

- Sets up an Amazon SQS queue

- Uses an existing queue if one is provided, otherwise creates a new one
- Adds an Interface Endpoint to the VPC for SQS (the service by default runs in Isolated or Private subnets)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:

@aws-solutions-constructs/aws-fargate-sqs

aws-fargate-ssmstringparameter

STABILITY EXPERIMENTAL

All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_fargate_ssmstringparameter`
Typescript	`@aws-solutions-constructs/aws-fargate-ssmstringparameter`
Java	`software.amazon.awsconstructs.services.fargatessmstringparameter`

Overview

This AWS Solutions Construct implements an AWS Fargate service that can read/write to an AWS Systems Manager String Parameter

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { FargateToSsmstringparameter, FargateToSsmstringparameterProps } from '@aws-solutions-constructs/aws-fargate-ssmstringparameter';

const constructProps: FargateToSsmstringparameterProps = {
  publicApi: true,
  ecrRepositoryArn: "arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-repo",
  stringParameterProps: { stringValue: "test-string-value" }
};

new FargateToSsmstringparameter(stack, 'test-construct', constructProps);
```

Python

```
from aws_solutions_constructs.aws_fargate_ssmstringparameter import
 FargateToSsmstringparameter, FargateToSsmstringparameterProps
from aws_cdk import (
    Stack,
    aws_ssm as ssm
)
from constructs import Construct

FargateToSsmstringparameter(self, 'test_construct',
            public_api=True,
            ecr_repository_arn="arn:aws:ecr:us-east-1:123456789012:repository/your-
ecr-repo",
            string_parameter_props=ssm.StringParameterProps(
              string_value="test-string-value"))
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.ssm.*;
import software.amazon.awsconstructs.services.fargatessmstringparameter.*;

new FargateToSsmstringparameter(this, "test-construct", new
 FargateToSsmstringparameterProps.Builder()
        .publicApi(true)
        .ecrRepositoryArn("arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-
repo")
        .stringParameterProps(new StringParameterProps.Builder()
                        .stringValue("test-string-value")
                        .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
publicApi	`boolean`	Whether the construct is deploying a private or public API. This has implications for the VPC.
vpcProps?	[`ec2.VpcProps`]()	Optional custom properties for a VPC the construct will create. This VPC will be used by any Private Hosted Zone the construct creates (that's why loadBalancerProps and privateHostedZoneProps can't include a VPC). Providing both this and existingVpc is an error.
existingVpc?	[`ec2.IVpc`]()	An existing VPC in which to deploy the construct. Providing both this and vpcProps is an error. If the client provides an existing load balancer and/or existing Private Hosted Zone, those constructs must exist in this VPC.
clusterProps?	[`ecs.ClusterProps`]()	Optional properties to create a new ECS cluster. To provide an existing cluster, use the cluster attribute of fargateServiceProp s.
ecrRepositoryArn?	`string`	The arn of an ECR Repository containing the image to use to generate the containers. Either this or the image property

Name	Type	Description
		of containerDefinitionProps must be provided. format: arn:aws:ecr:*region*:*account number*:repository/*Repository Name*
ecrImageVersion?	`string`	The version of the image to use from the repository. Defaults to "Latest"
containerDefinitionProps?	[`ecs.ContainerDefinitionProps | any`]()	Optional props to define the container created for the Fargate Service (defaults found in fargate-defaults.ts)
fargateTaskDefinitionProps?	[`ecs.FargateTaskDefinitionProps | any`]()	Optional props to define the Fargate Task Definition for this construct (defaults found in fargate-defaults.ts)
fargateServiceProps?	[`ecs.FargateServiceProps | any`]()	Optional values to override default Fargate Task definition properties (fargate-defaults.ts). The construct will default to launching the service is the most isolated subnets available (precedence: Isolated, Private and Public). Override those and other defaults here.

Name	Type	Description
existingFargateSer viceObject?	ecs.FargateService	A Fargate Service already instantiated (probably by another Solutions Construct). If this is specified, then no props defining a new service can be provided, including: ecrImageVersion, container DefinitionProps, fargateTa skDefinitionProps, ecrReposi toryArn, fargateServiceProps, clusterProps
existingContainerD efinitionObject?	ecs.ContainerDefin ition	A container definition already instantiated as part of a Fargate service. This must be the container in the existingF argateServiceObject
existingStringPara meterObj?	ssm.StringParameter	Existing instance of SSM String parameter object, providing both this and `stringPar ameterProps` will cause an error
stringParameterProps?	ssm.StringParamete rProps	Optional user provided props to override the default props for SSM String parameter. If existingStringParameterObj is not set stringParameterProps is required. The only supported ssm.StringParamete rProps.type is STRING if a different value is provided it will be overridden.

Name	Type	Description
stringParameterPer missions?	`string`	Optional SSM String parameter permissions to grant to the Fargate service. One of the following may be specified: "Read", "ReadWrite".
stringParameterEnv ironmentVariableName?	`string`	Optional Name for the container environment variable set to the SSM parameter name. Default: SSM_STRIN G_PARAMETER_NAME

Pattern Properties

Name	Type	Description
vpc	`ec2.IVpc`	The VPC used by the construct (whether created by the construct or provided by the client)
service	`ecs.FargateService`	The AWS Fargate service used by this construct (whether created by this construct or passed to this construct at initialization)
container	`ecs.ContainerDefin ition`	The container associated with the AWS Fargate service in the service property.
stringParameter	`ssm.StringParameter`	Returns an instance of `ssm.StringParameter` created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

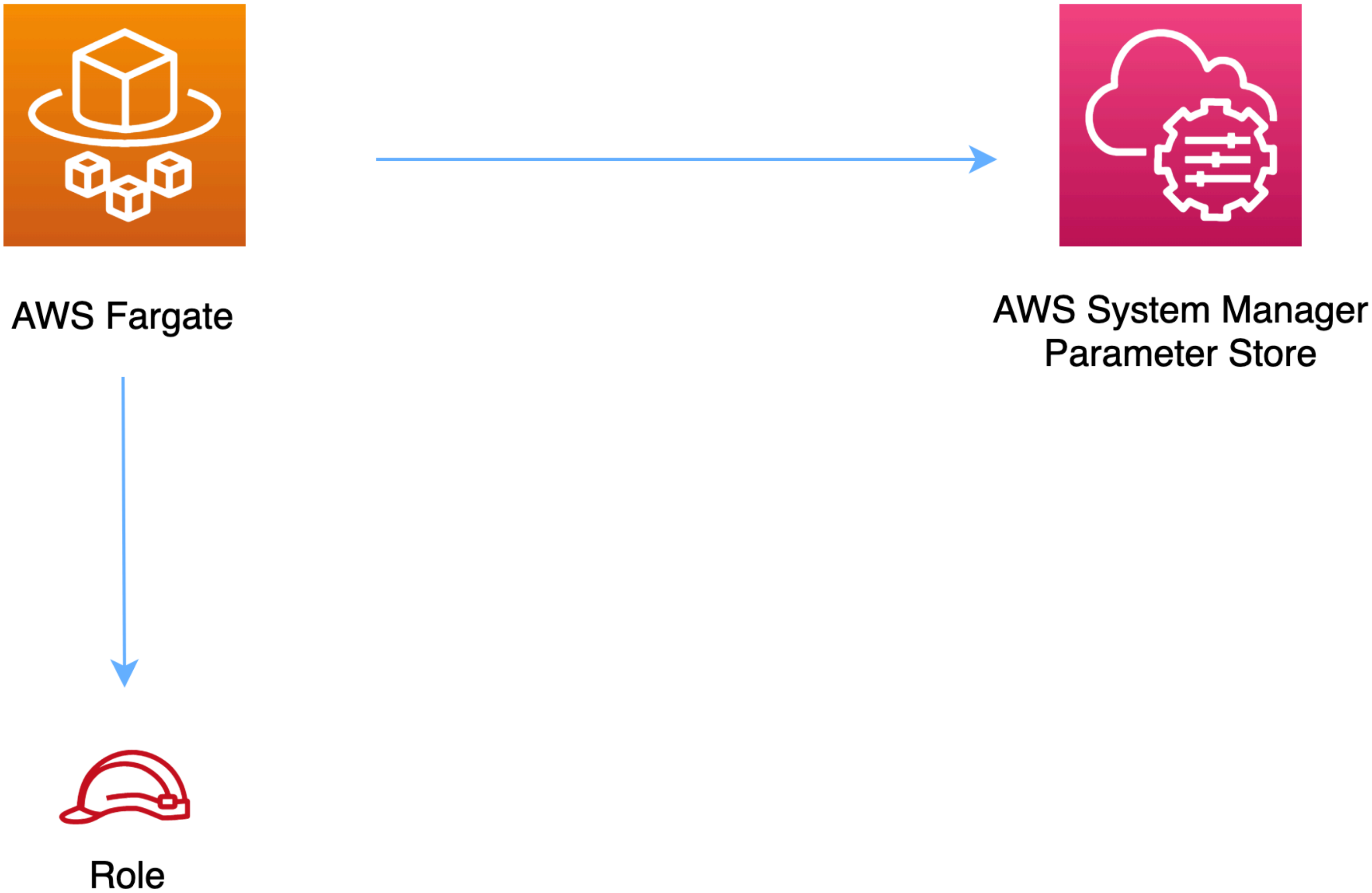
AWS Fargate Service

- Sets up an AWS Fargate service
  - Uses the existing service if provided
  - Creates a new service if none provided.
    - Service will run in isolated subnets if available, then private subnets if available and finally public subnets
  - Adds environment variables to the container with the ARN and Name of the SSM parameter
  - Add permissions to the container IAM role allowing it to read/write to the SSM parameter

AWS SSM String Parameter

- Sets up an AWS SSM String Parameter
  - Uses an existing parameter if one is provided, otherwise creates a new one
- Adds an Interface Endpoint to the VPC for SSM parameter (the service by default runs in Isolated or Private subnets)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-fargate-ssmstringparameter](https://github.com/awslabs/aws-solutions-constructs/tree/main/source/patterns/%40aws-solutions-constructs/aws-fargate-ssmstringparameter)

aws-fargate-stepfunctions

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic

[Versioning](#) model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_fargate_stepfunctions`
Typescript	`@aws-solutions-constructs/aws-fargate-stepfunctions`
Java	`software.amazon.awsconstructs.services.fargatestepfunctions`

Overview

This AWS Solutions Construct implements an AWS Fargate service that can execute an AWS Step Functions state machine

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { FargateToStepfunctions, FargateToStepfunctionsProps } from '@aws-solutions-constructs/aws-fargate-stepfunctions';
import * as stepfunctions from 'aws-cdk-lib/aws-stepfunctions';

const startState = new stepfunctions.Pass(this, 'StartState');

const constructProps: FargateToStepfunctionsProps = {
  publicApi: true,
  ecrRepositoryArn: "arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-repo",
  stateMachineProps: {
      definition: startState
    }
};
```

```
new FargateToStepfunctions(this, 'test-construct', constructProps);
```

Python

```
from aws_solutions_constructs.aws_fargate_stepfunctions import
 FargateToStepfunctions, FargateToStepfunctionsProps
from aws_cdk import (
    aws_stepfunctions as stepfunctions,
    Stack
)
from constructs import Construct

start_state = stepfunctions.Pass(self, 'start_state')

FargateToStepfunctions(self, 'test_construct',
            public_api=True,
            ecr_repository_arn="arn:aws:ecr:us-east-1:123456789012:repository/your-
ecr-repo",
            state_machine_props=stepfunctions.StateMachineProps(
              definition=start_state))
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.fargatestepfunctions.*;
import software.amazon.awscdk.services.stepfunctions.*;

start_state = stepfunctions.Pass(self, 'start_state')

new FargateToStepfunctions(this, "test-construct", new
 FargateToStepfunctionsProps.Builder()
        .publicApi(true)
        .ecrRepositoryArn("arn:aws:ecr:us-east-1:123456789012:repository/your-ecr-
repo")
        .stateMachineProps(new StateMachineProps.Builder()
                        .definition(startState)
                        .build()
        .build());
```

Pattern Construct Props

Name	Type	Description
publicApi	`boolean`	Whether the construct is deploying a private or public API. This has implications for the VPC.
vpcProps?	[`ec2.VpcProps`]()	Optional custom properties for a VPC the construct will create. This VPC will be used by any Private Hosted Zone the construct creates (that's why loadBalancerProps and privateHostedZoneProps can't include a VPC). Providing both this and existingVpc is an error.
existingVpc?	[`ec2.IVpc`]()	An existing VPC in which to deploy the construct. Providing both this and vpcProps is an error. If the client provides an existing load balancer and/or existing Private Hosted Zone, those constructs must exist in this VPC.
clusterProps?	[`ecs.ClusterProps`]()	Optional properties to create a new ECS cluster. To provide an existing cluster, use the cluster attribute of fargateServiceProp s.
ecrRepositoryArn?	`string`	The arn of an ECR Repository containing the image to use to generate the containers. Either this or the image property

Name	Type	Description
		of containerDefinitionProps must be provided. format: arn:aws:ecr:*region*:*account number*:repository/*Repository Name*
ecrImageVersion?	`string`	The version of the image to use from the repository. Defaults to "Latest"
containerDefinitionProps?	[`ecs.ContainerDefinitionProps | any`]()	Optional props to define the container created for the Fargate Service (defaults found in fargate-defaults.ts)
fargateTaskDefinitionProps?	[`ecs.FargateTaskDefinitionProps | any`]()	Optional props to define the Fargate Task Definition for this construct (defaults found in fargate-defaults.ts)
fargateServiceProps?	[`ecs.FargateServiceProps | any`]()	Optional values to override default Fargate Task definition properties (fargate-defaults.ts). The construct will default to launching the service is the most isolated subnets available (precedence: Isolated, Private and Public). Override those and other defaults here.

Name	Type	Description
existingFargateSer viceObject?	ecs.FargateService	A Fargate Service already instantiated (probably by another Solutions Construct). If this is specified, then no props defining a new service can be provided, including: ecrImageVersion, container DefinitionProps, fargateTa skDefinitionProps, ecrReposi toryArn, fargateServiceProps, clusterProps
existingContainerD efinitionObject?	ecs.ContainerDefin ition	A container definition already instantiated as part of a Fargate service. This must be the container in the existingF argateServiceObject
stateMachineProps	sfn.StateMachineProps	User provided props to override the default props for sfn.State Machine.
createCloudWatchAlarms?	boolean	Whether to create recommend ed CloudWatch alarms. Default is true.
logGroupProps?	logs.LogGroupProps	Optional user provided props to override the default props for for the CloudWatchLogs LogGroup.
stateMachineEnviro nmentVariableName?	string	Optional Name for the container environment variable set to the ARN of the state machine. Default: STATE_MAC HINE_ARN

Pattern Properties

Name	Type	Description
vpc	ec2.IVpc	The VPC used by the construct (whether created by the construct or provided by the client)
service	ecs.FargateService	The AWS Fargate service used by this construct (whether created by this construct or passed to this construct at initialization)
container	ecs.ContainerDefinition	The container associated with the AWS Fargate service in the service property.
stateMachine	sfn.StateMachine	Returns an instance of `sfn.StateMachine` created by the construct.
stateMachineLogGroup	logs.ILogGroup	Returns an instance of the `logs.ILogGroup` created by the construct for StateMachine.
cloudwatchAlarms?	cloudwatch.Alarm[]	Returns a list of `cloudwatch.Alarm` created by the construct.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

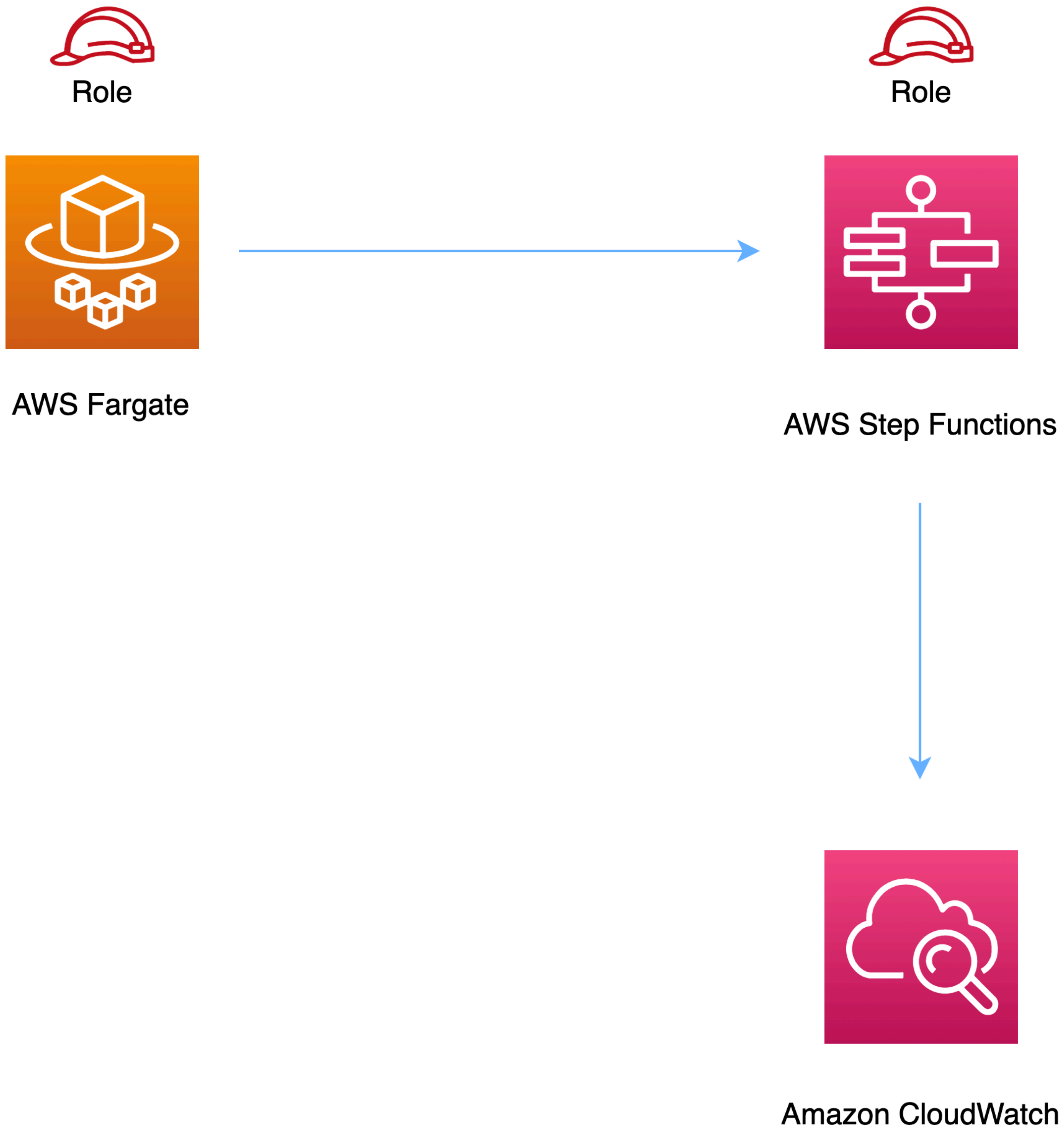
AWS Fargate Service

- Sets up an AWS Fargate service
  - Uses the existing service if provided
  - Creates a new service if none provided.
    - Service will run in isolated subnets if available, then private subnets if available and finally public subnets
- Adds an environment variable to the container containing the ARN of the state machine
  - Default name is `STATE_MACHINE_ARN`
- Add permissions to the container IAM role allowing it to start the execution of a state machine

AWS Step Functions

- Sets up an AWS Step Functions state machine
  - Uses an existing state machine if one is provided, otherwise creates a new one
- Adds an Interface Endpoint to the VPC for Step Functions (the service by default runs in Isolated or Private subnets)
- Enables CloudWatch logging

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-fargate-stepfunctions

aws-iot-kinesisfirehose-s3

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_iot_kinesisfirehose_s3`
Typescript	`@aws-solutions-constructs/aws-iot-kinesisfirehose-s3`
Java	`software.amazon.awsconstructs.services.iotkinesisfirehoses3`

Overview

This AWS Solutions Construct implements an AWS IoT MQTT topic rule to send data to an Amazon Kinesis Data Firehose delivery stream connected to an Amazon S3 bucket.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { IotToKinesisFirehoseToS3Props, IotToKinesisFirehoseToS3 } from '@aws-
solutions-constructs/aws-iot-kinesisfirehose-s3';

const constructProps: IotToKinesisFirehoseToS3Props = {
    iotTopicRuleProps: {
        topicRulePayload: {
            ruleDisabled: false,
            description: "Persistent storage of connected vehicle telematics data",
            sql: "SELECT * FROM 'connectedcar/telemetry/#'",
            actions: []
        }
    }
};

new IotToKinesisFirehoseToS3(this, 'test-iot-firehose-s3', constructProps);
```

Python

```
from aws_solutions_constructs.aws_iot_kinesisfirehose_s3 import
 IotToKinesisFirehoseToS3Props, IotToKinesisFirehoseToS3
from aws_cdk import (
    aws_iot as iot,
    Stack
)
from constructs import Construct

IotToKinesisFirehoseToS3(self, 'test_iot_firehose_s3',
                        iot_topic_rule_props=iot.CfnTopicRuleProps(

 topic_rule_payload=iot.CfnTopicRule.TopicRulePayloadProperty(
                                rule_disabled=False,
                                description="Persistent storage of connected vehicle
 telematics data",
                                sql="SELECT * FROM 'connectedcar/telemetry/#'",
                                actions=[]
                            )
                        ))
```

Java

```
import software.constructs.Construct;
import java.util.List;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.iot.*;
import software.amazon.awscdk.services.iot.CfnTopicRule.TopicRulePayloadProperty;
import software.amazon.awsconstructs.services.iotkinesisfirehoses3.*;

new IotToKinesisFirehoseToS3(this, "test-iot-firehose-s3", new
 IotToKinesisFirehoseToS3Props.Builder()
        .iotTopicRuleProps(new CfnTopicRuleProps.Builder()
                .topicRulePayload(new TopicRulePayloadProperty.Builder()
                        .ruleDisabled(false)
                        .description("Persistent storage of connected vehicle
 telematics data")
                        .sql("SELECT * FROM 'connectedcar/telemetry/#'")
                        .actions(List.of())
                        .build())
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
iotTopicRuleProps	`iot.CfnTopicRuleProps`	User provided CfnTopicRuleProps to override the defaults
kinesisFirehoseProps?	`kinesisfirehose.CfnDeliveryStreamProps`	Optional user provided props to override the default props for Kinesis Firehose Delivery Stream
existingBucketObj?	`s3.IBucket`	Existing instance of S3 Bucket object, providing both this and

Name	Type	Description
		`bucketProps` will cause an error.
bucketProps?	[`s3.BucketProps`]()	User provided props to override the default props for the S3 Bucket. If this is provided, then also providing bucketProps is an error.
logGroupProps?	[`logs.LogGroupProps`]()	User provided props to override the default props for for the CloudWatchLogs LogGroup.
loggingBucketProps?	[`s3.BucketProps`]()	Optional user provided props to override the default props for the S3 Logging Bucket.
logS3AccessLogs?	boolean	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with associated storage costs for the logs. Enabling Access Logging is a best practice. default - true

Pattern Properties

Name	Type	Description
kinesisFirehose	[`kinesisfirehose.CfnDeliveryStream`]()	Returns an instance of kinesisfirehose.CfnDeliveryStream created by the construct
s3Bucket?	[`s3.Bucket`]()	Returns an instance of s3.Bucket created by the construct

Name	Type	Description
s3LoggingBucket?	`s3.Bucket`	Returns an instance of s3.Bucket created by the construct as the logging bucket for the primary bucket.
iotTopicRule	`iot.CfnTopicRule`	Returns an instance of iot.CfnTopicRule created by the construct
iotActionsRole	`iam.Role`	Returns an instance of the iam.Role created by the construct for IoT Rule
kinesisFirehoseRole	`iam.Role`	Returns an instance of the iam.Role created by the construct for Kinesis Data Firehose delivery stream
kinesisFirehoseLogGroup	`logs.LogGroup`	Returns an instance of the LogGroup created by the construct for Kinesis Data Firehose delivery stream
s3BucketInterface	`s3.IBucket`	Returns an instance of s3.IBucket created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon IoT Rule

- Configure least privilege access IAM role for Amazon IoT

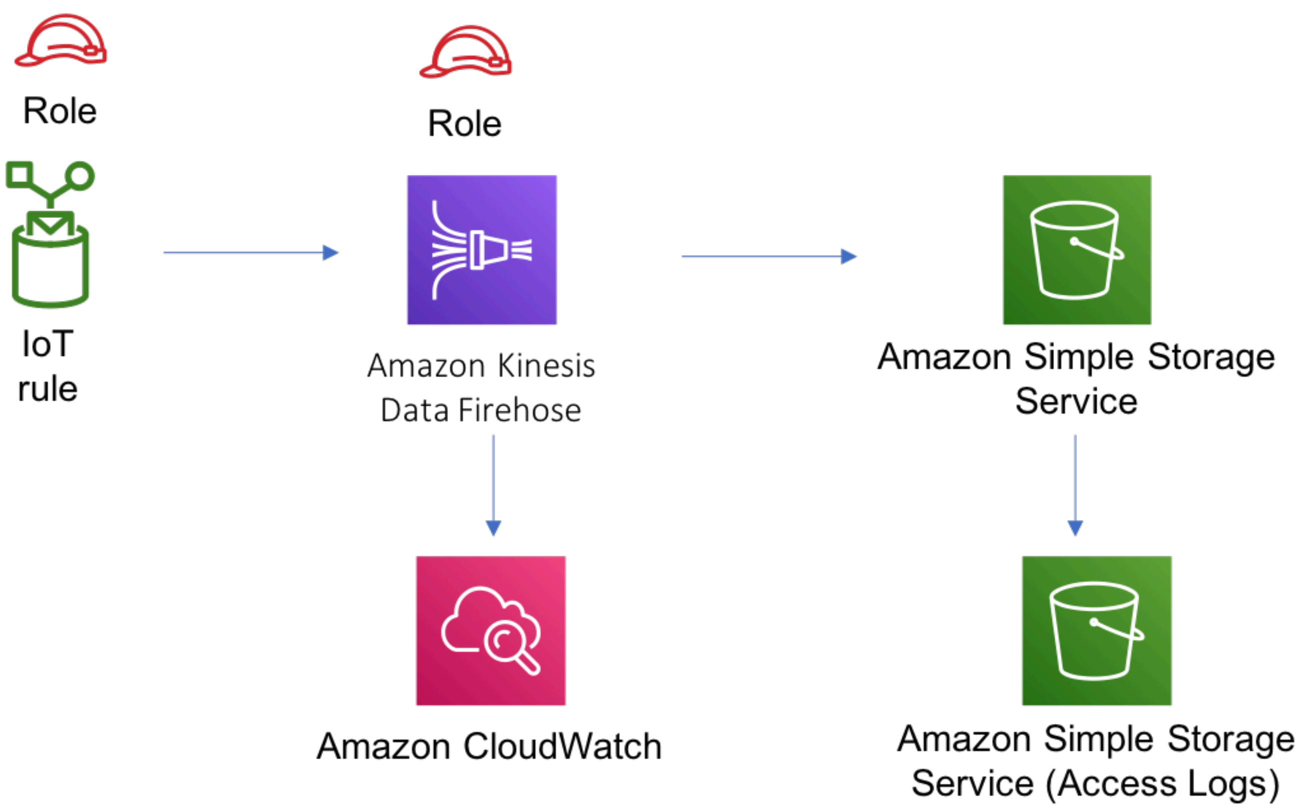
Amazon Kinesis Firehose

- Enable CloudWatch logging for Kinesis Firehose
- Configure least privilege access IAM role for Amazon Kinesis Firehose

Amazon S3 Bucket

- Configure Access logging for S3 Bucket
- Enable server-side encryption for S3 Bucket using AWS managed KMS Key
- Enforce encryption of data in transit
- Turn on the versioning for S3 Bucket
- Don't allow public access for S3 Bucket
- Retain the S3 Bucket when deleting the CloudFormation stack
- Applies Lifecycle rule to move noncurrent object versions to Glacier storage after 90 days

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-iot-kinesisfirehose-s3

aws-iot-kinesisstreams

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_iot_kinesisstreams`
Typescript	`@aws-solutions-constructs/aws-iot-kinesisstreams`
Java	`software.amazon.awsconstructs.services.iotkinesisstreams`

Overview

This AWS Solutions Construct implements an AWS IoT MQTT topic rule to send data to an Amazon Kinesis Data Stream.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { IotToKinesisStreamsProps, IotToKinesisStreams } from '@aws-solutions-constructs/aws-iot-kinesisstreams';

const constructProps: IotToKinesisStreamsProps = {
  iotTopicRuleProps: {
    topicRulePayload: {
      ruleDisabled: false,
      description: "Sends data to kinesis data stream",
      sql: "SELECT * FROM 'solutions/construct'",
      actions: []
    }
  }
};

new IotToKinesisStreams(this, 'test-iot-kinesisstreams', constructProps);
```

Python

```
from aws_solutions_constructs.aws_iot_kinesisstreams import
 IotToKinesisStreamsProps, IotToKinesisStreams
from aws_cdk import (
    aws_iot as iot,
    Stack
)
from constructs import Construct

IotToKinesisStreams(self, 'test-iot-kinesisstreams',
                    iot_topic_rule_props=iot.CfnTopicRuleProps(

 topic_rule_payload=iot.CfnTopicRule.TopicRulePayloadProperty(
                            rule_disabled=False,
                            description="Sends data to kinesis data stream",
                            sql="SELECT * FROM 'solutions/construct'",
                            actions=[]
                        )
                    ))
```

Java

```
import software.constructs.Construct;
import java.util.List;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.iot.*;
import software.amazon.awscdk.services.iot.CfnTopicRule.TopicRulePayloadProperty;
import software.amazon.awsconstructs.services.iotkinesisstreams.*;

new IotToKinesisStreams(this, "test-iot-kinesisstreams", new
 IotToKinesisStreamsProps.Builder()
        .iotTopicRuleProps(new CfnTopicRuleProps.Builder()
                .topicRulePayload(new TopicRulePayloadProperty.Builder()
                        .ruleDisabled(false)
                        .description("Sends data to kinesis data stream")
                        .sql("SELECT * FROM 'solutions/construct'")
                        .actions(List.of())
                        .build())
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
iotTopicRuleProps	iot.CfnTopicRuleProps	User provided CfnTopicRuleProps to override the defaults
existingStreamObj?	kinesis.Stream	Existing instance of Kinesis Stream, providing both this and `kinesisStreamProps` will cause an error.
kinesisStreamProps?	kinesis.StreamProps	Optional user-provided props to override the default props for the Kinesis data stream, providing both this and

Name	Type	Description
		`existingStreamObj` will cause an error
createCloudWatchAlarms	`boolean`	Whether to create recommended CloudWatch alarms for Kinesis Data Stream. Default value is set to `true`

Pattern Properties

Name	Type	Description
iotTopicRule	[`iot.CfnTopicRule`]()	Returns an instance of iot.CfnTopicRule created by the construct
iotActionsRole	[`iam.Role`]()	Returns an instance of the iam.Role created by the construct for IoT Rule
kinesisStream	[`kinesis.Stream`]()	Returns an instance of the Kinesis stream created by the construct.
cloudwatchAlarms?	[`cloudwatch.Alarm[]`]()	Returns an array of recommended CloudWatch Alarms created by the construct for Kinesis Data stream

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

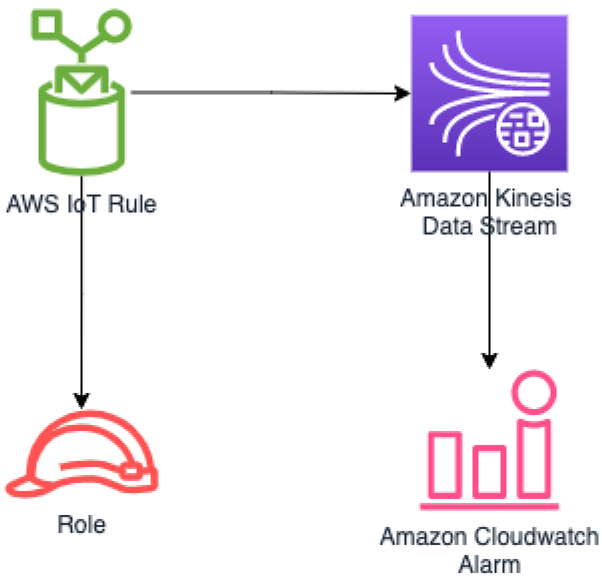
Amazon IoT Rule

- Configure least privilege access IAM role for Amazon IoT Rule

Amazon Kinesis Data Stream

- Configure recommended CloudWatch Alarms for Amazon Kinesis Data Stream
- Configure least privilege access IAM role for Amazon Kinesis Data Stream

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:

@aws-solutions-constructs/aws-iot-kinesisstreams

aws-iot-lambda-dynamodb

STABILITY EXPERIMENTAL

All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_iot_lambda_dynamodb
Typescript	@aws-solutions-constructs/aws-iot-lambda-dynamodb
Java	software.amazon.awsconstructs.services.iotlambdadynamodb

Overview

This AWS Solutions Construct implements an AWS IoT topic rule, an AWS Lambda function and Amazon DynamoDB table with the least privileged permissions.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { IotToLambdaToDynamoDBProps, IotToLambdaToDynamoDB } from '@aws-solutions-constructs/aws-iot-lambda-dynamodb';
import * as lambda from 'aws-cdk-lib/aws-lambda';

const constructProps: IotToLambdaToDynamoDBProps = {
  lambdaFunctionProps: {
      code: lambda.Code.fromAsset(`lambda`),
      runtime: lambda.Runtime.NODEJS_16_X,
      handler: 'index.handler'
  },
  iotTopicRuleProps: {
      topicRulePayload: {
```

```
          ruleDisabled: false,
          description: "Processing of DTC messages from the AWS Connected Vehicle
 Solution.",
          sql: "SELECT * FROM 'connectedcar/dtc/#'",
          actions: []
      }
  }
};

new IotToLambdaToDynamoDB(this, 'test-iot-lambda-dynamodb-stack', constructProps);
```

Python

```
from aws_solutions_constructs.aws_iot_lambda_dynamodb import IotToLambdaToDynamoDB
from aws_cdk import (
    aws_iot as iot,
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

IotToLambdaToDynamoDB(self, 'test-iot-lambda-dynamodb-stack',
            lambda_function_props=_lambda.FunctionProps(
                code=_lambda.Code.from_asset('lambda'),
                runtime=_lambda.Runtime.PYTHON_3_9,
                handler='index.handler'
            ),
            iot_topic_rule_props=iot.CfnTopicRuleProps(
                topic_rule_payload=iot.CfnTopicRule.TopicRulePayloadProperty(
                    rule_disabled=False,
                    description="Processing of DTC messages from the AWS Connected
 Vehicle Solution.",
                    sql="SELECT * FROM 'connectedcar/dtc/#'",
                    actions=[]
                )
            ))
```

Java

```
import software.constructs.Construct;
import java.util.List;
```

```
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awscdk.services.iot.*;
import software.amazon.awscdk.services.iot.CfnTopicRule.TopicRulePayloadProperty;
import software.amazon.awsconstructs.services.iotlambdadynamodb.*;

new IotToLambdaToDynamoDB(this, "test-iot-lambda-dynamodb-stack", new
 IotToLambdaToDynamoDBProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .iotTopicRuleProps(new CfnTopicRuleProps.Builder()
                .topicRulePayload(new TopicRulePayloadProperty.Builder()
                        .ruleDisabled(false)
                        .description("Processing of DTC messages from the AWS
 Connected Vehicle Solution.")
                        .sql("SELECT * FROM 'connectedcar/dtc/#'")
                        .actions(List.of())
                        .build())
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.

Name	Type	Description
iotTopicRuleProps	`iot.CfnTopicRuleProps`	User provided props to override the default props
dynamoTableProps?	`dynamodb.TableProps`	Optional user provided props to override the default props for DynamoDB Table
tablePermissions?	`string`	Optional table permissions to grant to the Lambda function. One of the following may be specified: `All`, `Read`, `ReadWrite` , `Write`.
existingTableObj?	`dynamodb.Table`	Existing instance of DynamoDB table object, providing both this and `dynamoTableProps` will cause an error.
tableEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the name of the DynamoDB table. Default: DDB_TABLE_NAME

Name	Type	Description
existingVpc?	`ec2.IVpc`	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to access network resources and a Gateway Endpoint will be created in the VPC for Amazon DynamoDB. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the `ec2.Vpc.fromLookup()` method.
vpcProps?	`ec2.VpcProps`	Optional user-provided properties to override the default properties for the new VPC. `enableDnsHostnames`, `enableDnsSupport`, `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.

Name	Type	Description
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:

Pattern Properties

Name	Type	Description
iotTopicRule	`iot.CfnTopicRule`	Returns an instance of iot.CfnTopicRule created by the construct
lambdaFunction	`lambda.Function`	Returns an instance of lambda.Function created by the construct
dynamoTable	`dynamodb.Table`	Returns an instance of dynamodb.Table created by the construct
vpc?	`ec2.IVpc`	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon IoT Rule

- Configure least privilege access IAM role for Amazon IoT

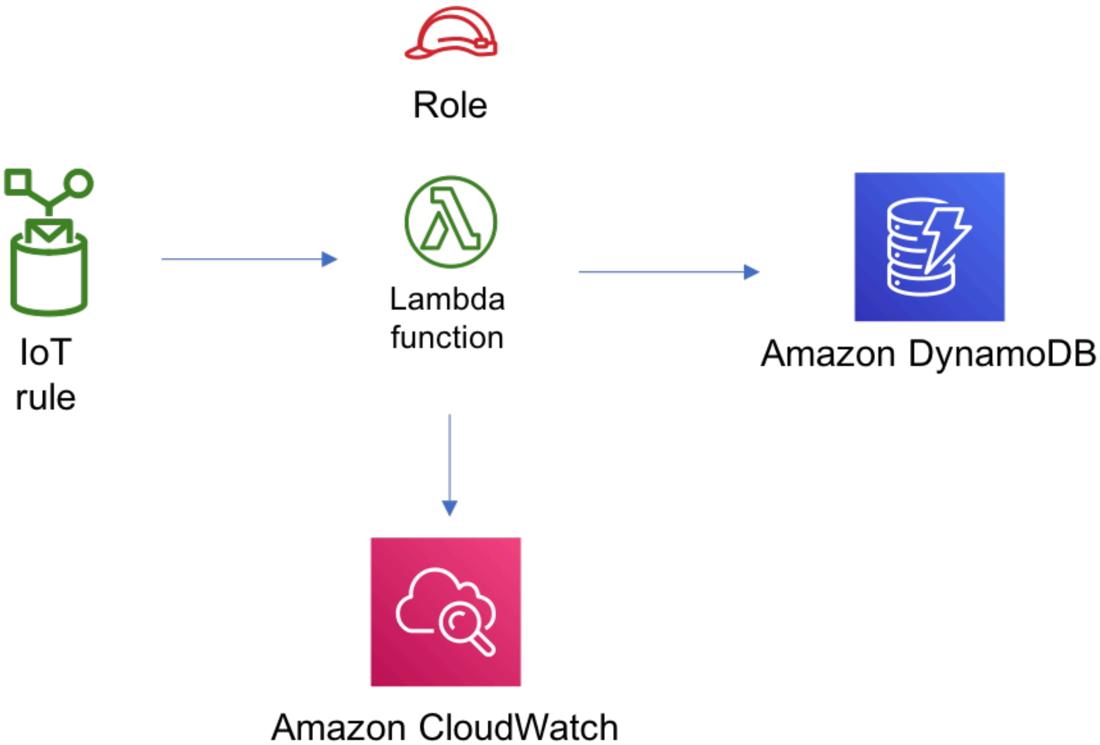
AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Amazon DynamoDB Table

- Set the billing mode for DynamoDB Table to On-Demand (Pay per request)
- Enable server-side encryption for DynamoDB Table using AWS managed KMS Key
- Creates a partition key called "id" for DynamoDB Table
- Retain the Table when deleting the CloudFormation stack
- Enable continuous backups and point-in-time recovery

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-iot-lambda-dynamodb

aws-iot-lambda

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aw s_iot_lambda
Typescript	@aws-solutions-constructs/aws-iot-lambda
Java	software.amazon.awsconstruc ts.services.iotlambda

Overview

This AWS Solutions Construct implements an AWS IoT MQTT topic rule and an AWS Lambda function pattern.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { IotToLambdaProps, IotToLambda } from '@aws-solutions-constructs/aws-iot-
lambda';
import * as lambda from 'aws-cdk-lib/aws-lambda';

const constructProps: IotToLambdaProps = {
  lambdaFunctionProps: {
    code: lambda.Code.fromAsset(`lambda`),
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler'
  },
  iotTopicRuleProps: {
    topicRulePayload: {
      ruleDisabled: false,
      description: "Processing of DTC messages from the AWS Connected Vehicle
 Solution.",
      sql: "SELECT * FROM 'connectedcar/dtc/#'",
      actions: []
    }
```

```
  }
};

new IotToLambda(this, 'test-iot-lambda-integration', constructProps);
```

Python

```
from aws_solutions_constructs.aws_iot_lambda import IotToLambdaProps, IotToLambda
from aws_cdk import (
    aws_iot as iot,
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

IotToLambda(self, 'test_iot_lambda',
            lambda_function_props=_lambda.FunctionProps(
                code=_lambda.Code.from_asset('lambda'),
                runtime=_lambda.Runtime.PYTHON_3_9,
                handler='index.handler'
            ),
            iot_topic_rule_props=iot.CfnTopicRuleProps(
                topic_rule_payload=iot.CfnTopicRule.TopicRulePayloadProperty(
                    rule_disabled=False,
                    description="Sends data to kinesis data stream",
                    sql="SELECT * FROM 'solutions/construct'",
                    actions=[]
                )
            ))
```

Java

```
import software.constructs.Construct;
import java.util.List;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awscdk.services.iot.*;
import software.amazon.awscdk.services.iot.CfnTopicRule.TopicRulePayloadProperty;
```

```
import software.amazon.awsconstructs.services.iotlambda.*;

new IotToLambda(this, "test-iot-lambda-integration", new IotToLambdaProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .iotTopicRuleProps(new CfnTopicRuleProps.Builder()
                .topicRulePayload(new TopicRulePayloadProperty.Builder()
                        .ruleDisabled(false)
                        .description("Processing of DTC messages from the AWS
 Connected Vehicle Solution.")
                        .sql("SELECT * FROM 'connectedcar/dtc/#'")
                        .actions(List.of())
                        .build())
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
iotTopicRuleProps?	`iot.CfnTopicRuleProps`	User provided CfnTopicRuleProps to override the defaults

Pattern Properties

Name	Type	Description
iotTopicRule	`iot.CfnTopicRule`	Returns an instance of iot.CfnTopicRule created by the construct
lambdaFunction	`lambda.Function`	Returns an instance of lambda.Function created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

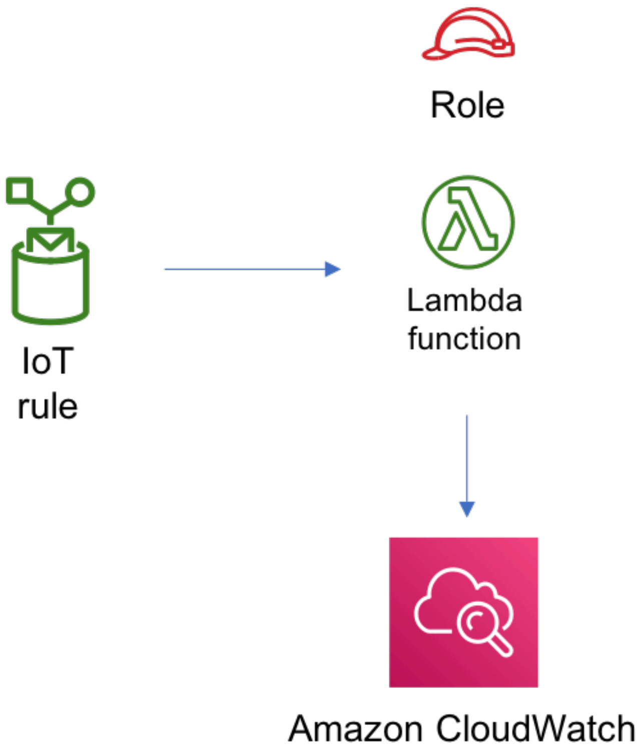
Amazon IoT Rule

- Configure least privilege access IAM role for Amazon IoT

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-iot-lambda](https://github.com/awslabs/aws-solutions-constructs/tree/master/source/patterns/%40aws-solutions-constructs/aws-iot-lambda)

aws-iot-s3

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the [Semantic Versioning](https://semver.org/) model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_iot_s3`
Typescript	`@aws-solutions-constructs/aws-iot-s3`
Java	`software.amazon.awsconstructs.services.iots3`

Overview

This AWS Solutions Construct implements an AWS IoT MQTT topic rule and an Amazon S3 Bucket pattern.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { IotToS3Props, IotToS3 } from '@aws-solutions-constructs/aws-iot-s3';

const constructProps: IotToS3Props = {
  iotTopicRuleProps: {
    topicRulePayload: {
      ruleDisabled: false,
      description: "Testing the IotToS3 Pattern",
      sql: "SELECT * FROM 'solutions/constructs'",
      actions: []
    }
  }
};

new IotToS3(this, 'test-iot-s3-integration', constructProps);
```

Python

```
from aws_solutions_constructs.aws_iot_s3 import IotToS3Props, IotToS3
from aws_cdk import (
    aws_iot as iot,
    Stack
)
from constructs import Construct


IotToS3(self, 'test_iot_s3',
        iot_topic_rule_props=iot.CfnTopicRuleProps(
            topic_rule_payload=iot.CfnTopicRule.TopicRulePayloadProperty(
                rule_disabled=False,
                description="Testing the IotToS3 Pattern",
                sql="SELECT * FROM 'solutions/constructs'",
                actions=[]
            )
        ))
```

Java

```
import software.constructs.Construct;
import java.util.List;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.iot.*;
import software.amazon.awscdk.services.iot.CfnTopicRule.TopicRulePayloadProperty;
import software.amazon.awsconstructs.services.iots3.*;

new IotToS3(this, "test_iot_s3", new IotToS3Props.Builder()
        .iotTopicRuleProps(new CfnTopicRuleProps.Builder()
                .topicRulePayload(new TopicRulePayloadProperty.Builder()
                        .ruleDisabled(false)
                        .description("Testing the IotToS3 Pattern")
                        .sql("SELECT * FROM 'solutions/constructs'")
                        .actions(List.of())
                        .build())
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingBucketInterface?	`s3.IBucket`	Existing S3 Bucket interface . Providing this property and `bucketProps` results in an error.
bucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the S3 Bucket. Providing this and `existingBucketObj` reults in an error.
loggingBucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the S3 Logging Bucket.
iotTopicRuleProps?	`iot.CfnTopicRuleProps`	User provided CfnTopicR uleProps to override the defaults.
s3Key	`string`	User provided s3Key to override the default (`${topic()}/${timestamp()}`) object key. Used to store messages matched by the IoT Rule.
logS3AccessLogs?	`boolean`	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with associated storage costs for the logs. Enabling Access Logging is a best practice. default - true

Pattern Properties

Name	Type	Description
s3Bucket?	s3.Bucket	Returns an instance of the S3 bucket created by the pattern. If an existingBucketInterface is provided in IotToS3Props, then this value will be undefined
s3BucketInterface?	s3.IBucket	Returns S3 Bucket interface created or used by the pattern. If an existingBucketInterface is provided in IotToS3Props, then only this value will be set and s3Bucket will be undefined . If the construct creates the bucket, then both properties will be set.
s3LoggingBucket?	s3.Bucket	Returns an instance of `s3.Bucket` created by the construct as the logging bucket for the primary bucket.
iotActionsRole	iam.Role	Returns an instance of `iam.Role` created by the construct, which allows IoT to publish messages to the S3 bucket.
iotTopicRule	iot.CfnTopicRule	Returns an instance of `iot.CfnTopicRule` created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon IoT Rule

- Configure an IoT Rule to send messages to the S3 Bucket

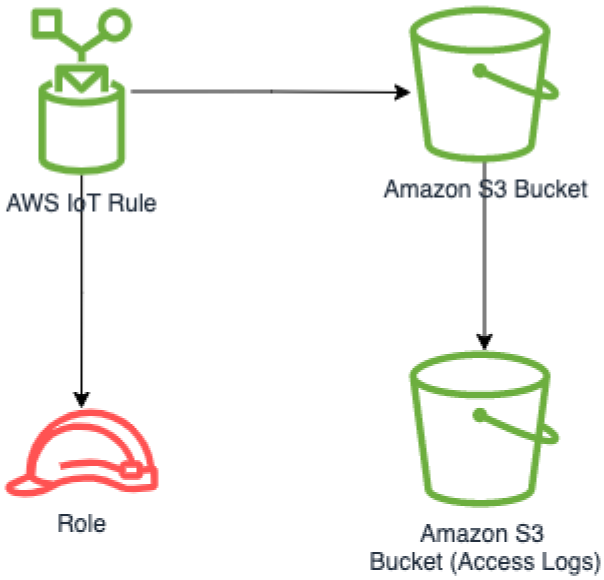
Amazon IAM Role

- Configure least privilege access IAM role for Amazon IoT to be able to publish messages to the S3 Bucket

Amazon S3 Bucket

- Configure Access logging for S3 Bucket
- Enable server-side encryption for S3 Bucket using AWS managed KMS Key
- Enforce encryption of data in transit
- Turn on the versioning for S3 Bucket
- Don’t allow public access for S3 Bucket
- Retain the S3 Bucket when deleting the CloudFormation stack
- Applies Lifecycle rule to move noncurrent object versions to Glacier storage after 90 days

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-iot-s3

aws-iot-sqs

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_iot_sqs`

Language	Package
Typescript	`@aws-solutions-constructs/aws-iot-sqs`
Java	`software.amazon.awsconstructs.services.iotsqs`

Overview

This AWS Solutions Construct implements an AWS IoT MQTT topic rule and an AWS SQS Queue pattern.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { IotToSqsProps, IotToSqs } from '@aws-solutions-constructs/aws-iot-sqs';

const constructProps: IotToSqsProps = {
  iotTopicRuleProps: {
    topicRulePayload: {
      ruleDisabled: false,
      description: "Testing the IotToSqs Pattern",
      sql: "SELECT * FROM 'iot/sqs/#'",
      actions: []
    }
  }
};

new IotToSqs(this, 'test-iot-sqs-integration', constructProps);
```

Python

```
from aws_solutions_constructs.aws_iot_sqs import IotToSqs
from aws_cdk import (
```

```
    aws_iot as iot,
    Stack
)
from constructs import Construct

IotToSqs(self, 'test_iot_sqs',
    iot_topic_rule_props=iot.CfnTopicRuleProps(
        topic_rule_payload=iot.CfnTopicRule.TopicRulePayloadProperty(
            rule_disabled=False,
            description="Testing the IotToSqs Pattern",
            sql="SELECT * FROM 'iot/sqs/#'",
            actions=[]
        )
    ))
```

Java

```
import software.constructs.Construct;
import java.util.List;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.iot.*;
import software.amazon.awscdk.services.iot.CfnTopicRule.TopicRulePayloadProperty;
import software.amazon.awsconstructs.services.iotsqs.*;

new IotToSqs(this, "test_iot_sqs", new IotToSqsProps.Builder()
        .iotTopicRuleProps(new CfnTopicRuleProps.Builder()
                .topicRulePayload(new TopicRulePayloadProperty.Builder()
                        .ruleDisabled(false)
                        .description("Testing the IotToSqs Pattern")
                        .sql("SELECT * FROM 'iot/sqs/#'")
                        .actions(List.of())
                        .build())
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
iotTopicRuleProps	`iot.CfnTopicRuleProps`	User provided CfnTopicRuleProps to override the defaults
existingQueueObj?	`sqs.Queue`	Existing instance of SQS queue object, providing both this and `queueProps` will cause an error.
queueProps?	`sqs.QueueProps`	User provided props to override the default props for the SQS queue.
deadLetterQueueProps?	`sqs.QueueProps`	Optional user provided properties for the dead letter queue.
deployDeadLetterQueue?	`boolean`	Whether to deploy a secondary queue to be used as a dead letter queue. Default `true`.
maxReceiveCount?	`number`	The number of times a message can be unsuccessfully dequeued before being moved to the dead-letter queue. Required field if `deployDeadLetterQueue =true`.
enableEncryptionWithCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the queue is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined:

Name	Type	Description
		queueProps.encryptionMaster Key, encryptionKey or encryptio nKeyProps.
encryptionKey?	`kms.Key`	An optional, imported encryptio n key to encrypt the SQS Queue with.
encryptionKeyProps?	`kms.KeyProps`	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SQS queue with.

Pattern Properties

Name	Type	Description
encryptionKey?	`kms.Key`	Returns an instance of `kms.Key` used for the SQS queue.
iotActionsRole	`iam.Role`	Returns an instance of `iam.Role` created by the construct, which allows IoT to publish messages to the SQS Queue
sqsQueue	`sqs.Queue`	Returns an instance of `sqs.Queue` created by the construct
deadLetterQueue?	`sqs.Queue`	Returns an instance of the dead-letter SQS queue created by the pattern.

Name	Type	Description
iotTopicRule	`iot.CfnTopicRule`	Returns an instance of `iot.CfnTopicRule` created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon IoT Rule

- Configure an IoT Rule to send messages to the SQS Queue

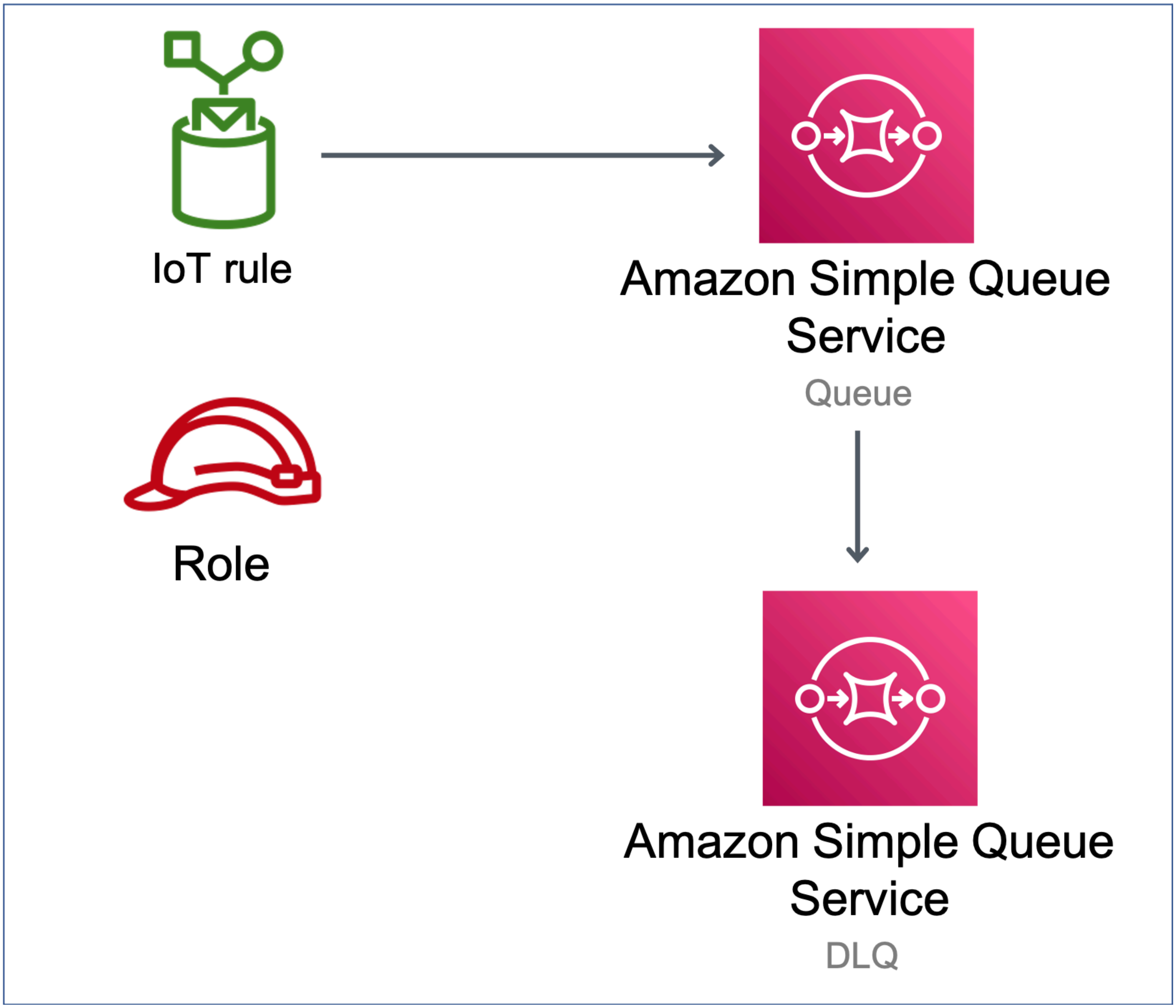
Amazon IAM Role

- Configure least privilege access IAM role for Amazon IoT to be able to publish messages to the SQS Queue

Amazon SQS Queue

- Deploy a dead-letter queue for the source queue.
- Enable server-side encryption for the source queue using a customer-managed AWS KMS key.
- Enforce encryption of data in transit.

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-iot-sqs

aws-kinesisfirehose-s3

CFN-RESOURCES STABLE

Language	Package
Python	aws_solutions_constructs.aws_kinesis_firehose_s3
Typescript	@aws-solutions-constructs/aws-kinesisfirehose-s3
Java	software.amazon.awsconstructs.services.kinesisfirehoses3

Overview

This AWS Solutions Construct implements an Amazon Kinesis Data Firehose delivery stream connected to an Amazon S3 bucket.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { KinesisFirehoseToS3 } from '@aws-solutions-constructs/aws-kinesisfirehose-s3';

new KinesisFirehoseToS3(this, 'test-firehose-s3', {});
```

Python

```
from aws_solutions_constructs.aws_kinesis_firehose_s3 import KinesisFirehoseToS3
from aws_cdk import Stack
from constructs import Construct
```

```
KinesisFirehoseToS3(self, 'test_firehose_s3')
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.kinesisfirehoses3.*;

new KinesisFirehoseToS3(this, "test_firehose_s3", new
 KinesisFirehoseToS3Props.Builder()
        .build());
```

Pattern Construct Props

Name	Type	Description
bucketProps?	s3.BucketProps	Optional user provided props to override the default props for the S3 Bucket.
existingBucketObj?	s3.IBucket	Optional existing instance of S3 Bucket. If this is provided, then also providing bucketProps is an error.
existingLoggingBucketObj?	s3.IBucket	Optional existing instance of logging S3 Bucket for the S3 Bucket created by the pattern.
kinesisFirehoseProps?	kinesisfirehose.CfnDeliveryStreamProps |any	Optional user provided props to override the default props for Kinesis Firehose Delivery Stream.
logGroupProps?	logs.LogGroupProps	Optional user provided props to override the default props

Name	Type	Description
		for for the CloudWatchLogs LogGroup.
loggingBucketProps?	s3.BucketProps	Optional user provided props to override the default props for the S3 Logging Bucket.
logS3AccessLogs?	boolean	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with associated storage costs for the logs. Enabling Access Logging is a best practice. default - true

Pattern Properties

Name	Type	Description
kinesisFirehose	kinesisfirehose.CfnDeliveryStream	Returns an instance of kinesisfirehose.CfnDeliveryStream created by the construct
kinesisFirehoseLogGroup	logs.LogGroup	Returns an instance of the logs.LogGroup created by the construct for Kinesis Data Firehose delivery stream
kinesisFirehoseRole	iam.Role	Returns an instance of the iam.Role created by the construct for Kinesis Data Firehose delivery stream
s3Bucket?	s3.Bucket	Returns an instance of s3.Bucket created by the construct

Name	Type	Description
s3LoggingBucket?	`s3.Bucket`	Returns an instance of s3.Bucket created by the construct as the logging bucket for the primary bucket
s3BucketInterface	`s3.IBucket`	Returns an instance of s3.IBucket created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

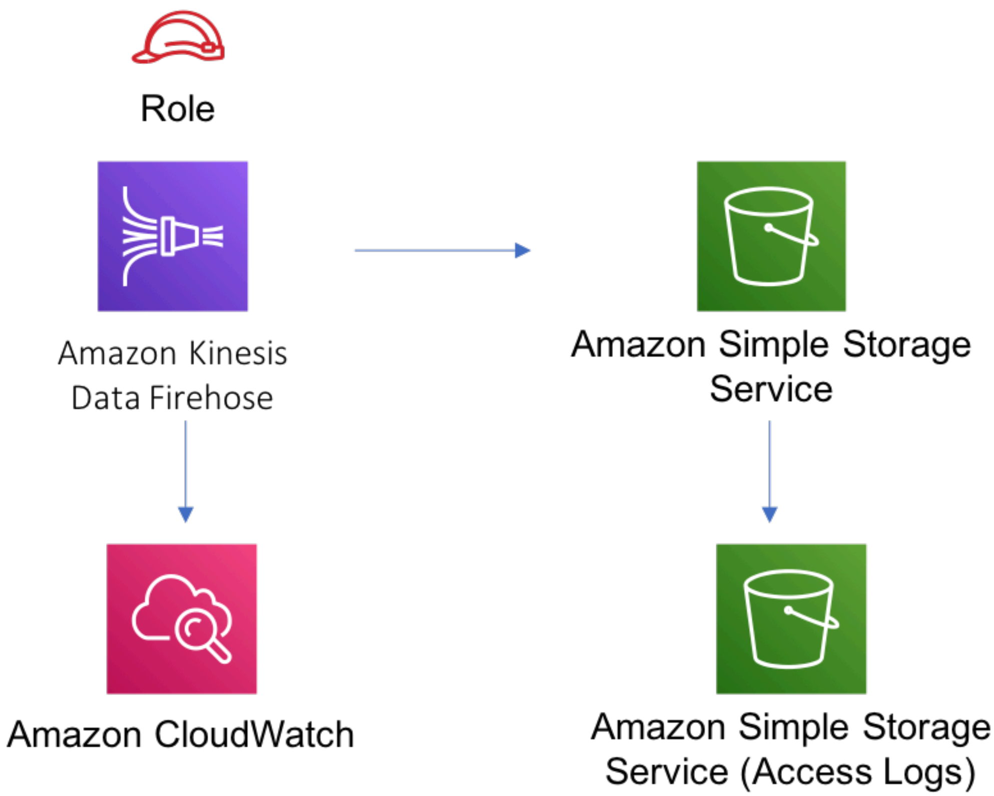
Amazon Kinesis Firehose

- Enable CloudWatch logging for Kinesis Firehose
- Configure least privilege access IAM role for Amazon Kinesis Firehose

Amazon S3 Bucket

- Configure Access logging for S3 Bucket
- Enable server-side encryption for S3 Bucket using AWS managed KMS Key
- Enforce encryption of data in transit
- Turn on the versioning for S3 Bucket
- Don’t allow public access for S3 Bucket
- Retain the S3 Bucket when deleting the CloudFormation stack
- Applies Lifecycle rule to move noncurrent object versions to Glacier storage after 90 days

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-kinesisfirehose-s3

aws-kinesisstreams-gluejob

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_kinesis_streams_gluejob
Typescript	@aws-solutions-constructs/aws-kinesisstreams-gluejob
Java	software.amazon.awsconstructs.services.kinesisstreamsgluejob

Overview

This AWS Solutions Construct deploys a Kinesis Stream and configures a AWS Glue Job to perform custom ETL transformation with the appropriate resources/properties for interaction and security. It also creates an S3 bucket where the python script for the AWS Glue Job can be uploaded.

Here is a minimal deployable pattern definition:

Typescript

```
import * as glue from "@aws-cdk/aws-glue";
import * as s3assets from "@aws-cdk/aws-s3-assets";
import { KinesisstreamsToGluejob } from "@aws-solutions-constructs/aws-
kinesisstreams-gluejob";

const fieldSchema: glue.CfnTable.ColumnProperty[] = [
  {
    name: "id",
    type: "int",
    comment: "Identifier for the record",
  },
  {
    name: "name",
    type: "string",
    comment: "Name for the record",
```

```
  },
  {
    name: "address",
    type: "string",
    comment: "Address for the record",
  },
  {
    name: "value",
    type: "int",
    comment: "Value for the record",
  },
];

const customEtlJob = new KinesisstreamsToGluejob(this, "CustomETL", {
  glueJobProps: {
    command: {
      name: "gluestreaming",
      pythonVersion: "3",
    },
  },
  fieldSchema: fieldSchema,
  etlCodeAsset: new s3assets.Asset(this, "ScriptLocation", {
    path: `${__dirname}/../etl/transform.py`,
  }),
});
```

Pattern Construct Props

Name	Type	Description
existir tream	`kinesis.Stream`	Existing instance of Kinesis Stream, providing both this and `kinesisStreamProps` will cause an error.
kinesi reamI ?	`kinesis.StreamProps`	Optional user-provided props to override the default props for the Kinesis stream.
glueJ ops?	`cfnJob.CfnJobProps`	User provided props to override the default props for the AWS Glue Job.

Name	Type	Description
existir lueJol	`cfnJob.CfnJob`	Existing instance of AWS Glue Job, providing both this and `glueJobProps` will cause an error.
fieldS ma?	`CfnTable.ColumnProperty[]`	User provided schema structure to create an AWS Glue Table.
existir able?	`CfnTable`	Existing instance of AWS Glue Table. If this is set, tableProps and fieldSchema are ignored.
tableF s?	`CfnTableProps`	User provided AWS Glue Table props to override default props used to create a Glue Table.
existir ataba	`CfnDatabase`	Existing instance of AWS Glue Database. If this is set, then databaseProps is ignored.
datab rops?	`CfnDatabaseProps`	User provided Glue Database Props to override the default props used to create the Glue Database.
outpu aStor	`SinkDataStoreProps`	User provided properties for S3 bucket that stores Glue Job output. Current datastore types supported is only S3.
create udWa arms?	`boolean`	Whether to create recommended CloudWatch alarms for Kinesis Data Stream. Default value is set to `true`.
etlCo set?	s3assets.Asset	User provided instance of the Asset class that represents the ETL code on the local filesystem

SinkDataStoreProps

Name	Type	Description
existingS 3Output cket?	Bucket	Existing instance of S3 bucket where the data should be written. Providing both this and `outputBucketProps` will cause an error.
outputBu ketProps	BucketProps	User provided bucket properties to create the S3 bucket to store the output from the AWS Glue Job.
datastor Type	SinkStoreType	Sink data store type.

SinkStoreType

Enumeration of data store types that could include S3, DynamoDB, DocumentDB, RDS or Redshift. Current construct implementation only supports S3, but potential to add other output types in the future.

Name	Type	Description
S3	`string`	S3 storage type

Pattern Properties

Name	Type	Description
kinesisStream	kinesis.Stream	Returns an instance of the Kinesis stream created or used by the pattern.

Name	Type	Description
glueJob	CfnJob	Returns an instance of AWS Glue Job created by the construct.
glueJobRole	iam.Role	Returns an instance of the IAM Role created by the construct for the Glue Job.
database	CfnDatabase	Returns an instance of AWS Glue Database created by the construct.
table	CfnTable	Returns an instance of the AWS Glue Table created by the construct
outputBucket?	s3.Bucket	Returns an instance of the output bucket created by the construct for the AWS Glue Job.
cloudwatchAlarms?	cloudwatch.Alarm[]	Returns an array of recommended CloudWatch Alarms created by the construct for Kinesis Data stream.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon Kinesis Stream

- Configure least privilege access IAM role for Kinesis Stream
- Enable server-side encryption for Kinesis Stream using AWS Managed KMS Key
- Deploy best practices CloudWatch Alarms for the Kinesis Stream

Glue Job

- Create a Glue Security Config that configures encryption for CloudWatch, Job Bookmarks, and S3. CloudWatch and Job Bookmarks are encrypted using AWS Managed KMS Key created for AWS Glue Service. The S3 bucket is configured with SSE-S3 encryption mode
- Configure service role policies that allow AWS Glue to read from Kinesis Data Streams

Glue Database

- Create an AWS Glue database. An AWS Glue Table will be added to the database. This table defines the schema for the records buffered in the Amazon Kinesis Data Streams

Glue Table

- Create an AWS Glue table. The table schema definition is based on the JSON structure of the records buffered in the Amazon Kinesis Data Streams

IAM Role

- A job execution role that has privileges to 1) read the ETL script from the S3 bucket location, 2) read records from the Kinesis Stream, and 3) execute the Glue Job

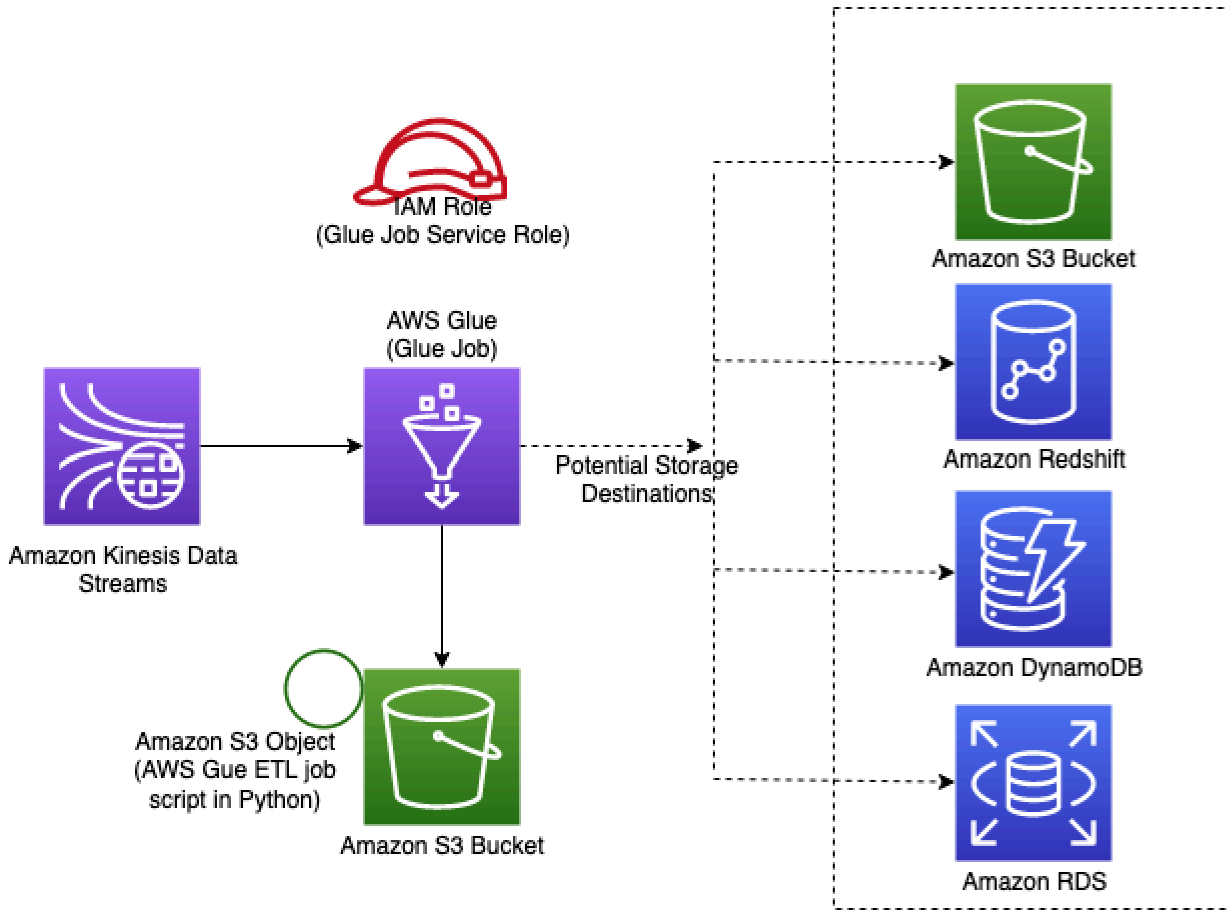
Output S3 Bucket

- An S3 bucket to store the output of the ETL transformation. This bucket will be passed as an argument to the created glue job so that it can be used in the ETL script to write data into it

Cloudwatch Alarms

- A CloudWatch Alarm to report when consumer application is reading data slower than expected
- A CloudWatch Alarm to report when consumer record processing is falling behind (to avoid risk of data loss due to record expiration)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-kinesisstreams-gluejob]

Reference Implementation

A sample use case which uses this pattern is available under `use_cases/aws-custom-glue-etl`.

aws-kinesisstreams-kinesisfirehose-s3

CFN-RESOURCES STABLE

Language	Package
Python	aws_solutions_constructs.aws_kinesisstreams_kinesisfirehose_s3
Typescript	@aws-solutions-constructs/aws-kinesis-streams-kinesis-firehose-s3
Java	software.amazon.awsconstructs.services.kinesisstreamskinesisfirehoses3

Overview

This AWS Solutions Construct implements an Amazon Kinesis Data Stream (KDS) connected to Amazon Kinesis Data Firehose (KDF) delivery stream connected to an Amazon S3 bucket.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { KinesisStreamsToKinesisFirehoseToS3 } from '@aws-solutions-constructs/aws-kinesisstreams-kinesisfirehose-s3';

new KinesisStreamsToKinesisFirehoseToS3(this, 'test-stream-firehose-s3', {});
```

Python

```
from aws_solutions_constructs.aws_kinesis_streams_kinesis_firehose_s3 import
 KinesisStreamsToKinesisFirehoseToS3
from aws_cdk import Stack
from constructs import Construct
```

```
KinesisStreamsToKinesisFirehoseToS3(self, 'test_stream_firehose_s3')
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.kinesisstreamskinesisfirehoses3.*;

new KinesisStreamsToKinesisFirehoseToS3(this, "test_stream_firehose_s3", new
 KinesisStreamsToKinesisFirehoseToS3Props.Builder()
        .build());
```

Pattern Construct Props

Name	Type	Description
bucketProps?	s3.BucketProps	Optional user provided props to override the default props for the S3 Bucket.
createCloudWatchAlarms?	boolean	Optional whether to create recommended CloudWatch alarms.
existingBucketObj?	s3.IBucket	Optional existing instance of S3 Bucket object. If this is provided, then also providing bucketProps is an error.
existingLoggingBucketObj?	s3.IBucket	Optional existing instance of logging S3 Bucket object for the S3 Bucket created by the pattern.
existingStreamObj?	kinesis.Stream	Optional existing instance of Kinesis Stream, providing

Name	Type	Description
		both this and `kinesisStreamProps` will cause an error.
kinesisFirehoseProps?	`kinesisfirehose.CfnDeliveryStreamProps` |any	Optional user provided props to override the default props for Kinesis Firehose Delivery Stream.
kinesisStreamProps?	`kinesis.StreamProps`	Optional user-provided props to override the default props for the Kinesis stream.
logGroupProps?	`logs.LogGroupProps`	Optional user provided props to override the default props for for the CloudWatchLogs LogGroup.
loggingBucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the S3 Logging Bucket.
logS3AccessLogs?	boolean	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with associated storage costs for the logs. Enabling Access Logging is a best practice. default - true

Pattern Properties

Name	Type	Description
cloudwatchAlarms?	cloudwatch.Alarm[]	Returns a list of cloudwatch.Alarm created by the construct
kinesisFirehose	kinesisfirehose.CfnDeliveryStream	Returns an instance of kinesisfirehose.CfnDeliveryStream created by the construct
kinesisFirehoseLogGroup	logs.LogGroup	Returns an instance of the logs.LogGroup created by the construct for Kinesis Data Firehose delivery stream
kinesisFirehoseRole	iam.Role	Returns an instance of the iam.Role created by the construct for Kinesis Data Firehose delivery stream
kinesisStream	kinesis.Stream	Returns an instance of the Kinesis stream created by the pattern
kinesisStreamRole	iam.Role	Returns an instance of the iam.Role created by the construct for Kinesis stream
s3Bucket?	s3.Bucket	Returns an instance of s3.Bucket created by the construct
s3LoggingBucket?	s3.Bucket	Returns an instance of s3.Bucket created by the construct as the logging bucket for the primary bucket

Name	Type	Description
s3BucketInterface	`s3.IBucket`	Returns an instance of s3.IBucke t created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon Kinesis Stream

- Configure least privilege access IAM role for Kinesis Stream
- Enable server-side encryption for Kinesis Stream using AWS Managed KMS Key
- Deploy best practices CloudWatch Alarms for the Kinesis Stream

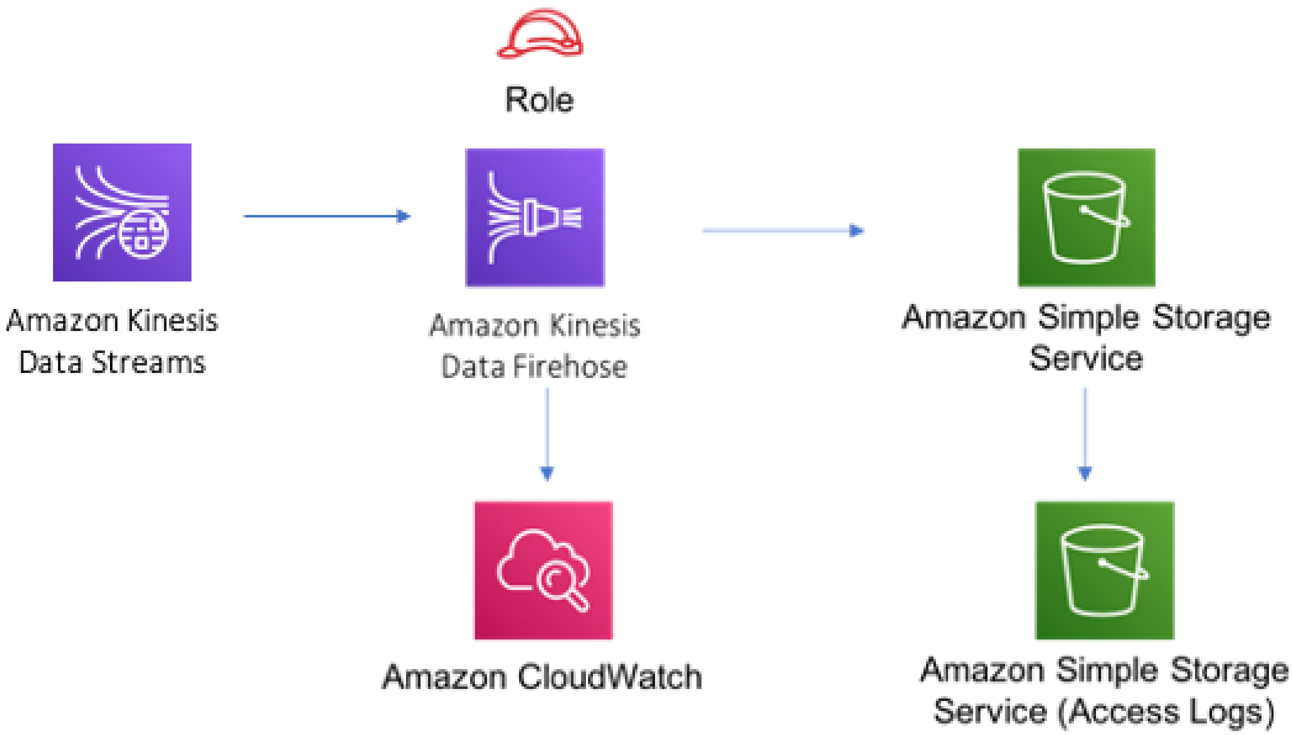
Amazon Kinesis Firehose

- Enable CloudWatch logging for Kinesis Firehose
- Configure least privilege access IAM role for Amazon Kinesis Firehose

Amazon S3 Bucket

- Configure Access logging for S3 Bucket
- Enable server-side encryption for S3 Bucket using AWS managed KMS Key
- Enforce encryption of data in transit
- Turn on the versioning for S3 Bucket
- Don't allow public access for S3 Bucket
- Retain the S3 Bucket when deleting the CloudFormation stack
- Applies Lifecycle rule to move noncurrent object versions to Glacier storage after 90 days

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-kinesisstreams-kinesisfirehose-s3

aws-kinesisstreams-lambda

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws-kinesis-streams-lambda`

Language	Package
Typescript	`@aws-solutions-constructs/aws-kinesisstreams-lambda`
Java	`software.amazon.awsconstructs.services.kinesisstreamslambda`

Overview

This AWS Solutions Construct deploys a Kinesis Stream and Lambda function with the appropriate resources/properties for interaction and security.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { KinesisStreamsToLambda } from '@aws-solutions-constructs/aws-
kinesisstreams-lambda';
import * as lambda from 'aws-cdk-lib/aws-lambda';

new KinesisStreamsToLambda(this, 'KinesisToLambdaPattern', {
  kinesisEventSourceProps: {
    startingPosition: lambda.StartingPosition.TRIM_HORIZON,
    batchSize: 1
  },
  lambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler',
    code: lambda.Code.fromAsset(`lambda`)
  }
});
```

Python

```
from aws_solutions_constructs.aws_kinesis_streams_lambda import
 KinesisStreamsToLambda
```

```
from aws_cdk import (
    aws_lambda as _lambda,
    aws_lambda_event_sources as sources,
    aws_kinesis as kinesis,
    Stack
)
from constructs import Construct

KinesisStreamsToLambda(self, 'KinesisToLambdaPattern',
                        kinesis_event_source_props=sources.KinesisEventSourceProps(
                            starting_position=_lambda.StartingPosition.TRIM_HORIZON,
                            batch_size=1
                        ),
                        lambda_function_props=_lambda.FunctionProps(
                            runtime=_lambda.Runtime.PYTHON_3_9,
                            handler='index.handler',
                            code=_lambda.Code.from_asset(
                                'lambda')
                        )
                        )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.eventsources.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.kinesisstreamslambda.*;

new KinesisStreamsToLambda(this, "KinesisToLambdaPattern", new
 KinesisStreamsToLambdaProps.Builder()
        .kinesisEventSourceProps(new KinesisEventSourceProps.Builder()
                .startingPosition(StartingPosition.TRIM_HORIZON)
                .batchSize(1)
                .build())
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
```

```
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
kinesisStreamProps?	`kinesis.StreamProps`	Optional user-provided props to override the default props for the Kinesis stream.
existingStreamObj?	`kinesis.Stream`	Existing instance of Kinesis Stream, providing both this and `kinesisStreamProps` will cause an error.
kinesisEventSourceProps?	`aws-lambda-event-sources.KinesisEventSourceProps`	Optional user-provided props to override the default props for the Lambda event source mapping.
createCloudWatchAlarms	`boolean`	Whether to create recommended CloudWatch alarms

Pattern Properties

Name	Type	Description
kinesisStream	`kinesis.Stream`	Returns an instance of the Kinesis stream created by the pattern.
lambdaFunction	`lambda.Function`	Returns an instance of the Lambda function created by the pattern.
kinesisStreamRole	`iam.Role`	Returns an instance of the iam.Role created by the construct for Kinesis stream.
cloudwatchAlarms?	`cloudwatch.Alarm[]`	Returns a list of cloudwatc h.Alarm created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon Kinesis Stream

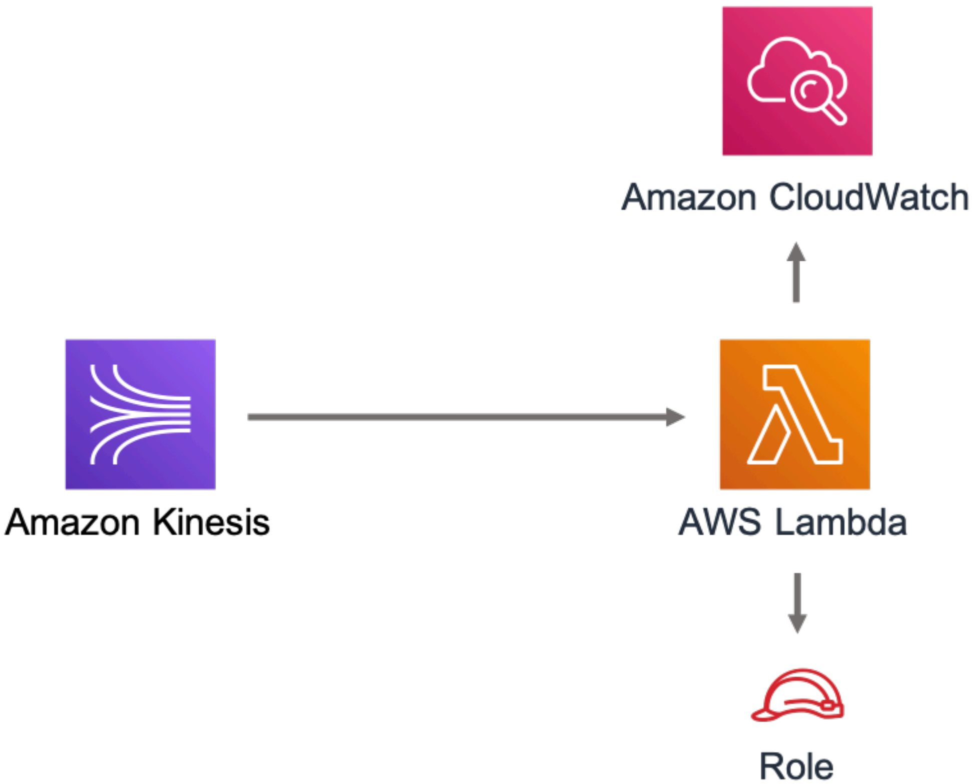
- Configure least privilege access IAM role for Kinesis Stream
- Enable server-side encryption for Kinesis Stream using AWS Managed KMS Key
- Deploy best practices CloudWatch Alarms for the Kinesis Stream

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing

- Enable Failure-Handling features like enable bisect on function Error, set defaults for Maximum Record Age (24 hours) & Maximum Retry Attempts (500) and deploy SQS dead-letter queue as destination on failure
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-kinesisstreams-lambda

aws-lambda-dynamodb

CFN-RESOURCES STABLE

Language	Package
Python	aws_solutions_constructs.aws_lambda_dynamodb
Typescript	@aws-solutions-constructs/aws-lambda-dynamodb
Java	software.amazon.awsconstructs.services.lambdadynamodb

Overview

This AWS Solutions Construct implements the AWS Lambda function and Amazon DynamoDB table with the least privileged permissions.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { LambdaToDynamoDBProps, LambdaToDynamoDB } from '@aws-solutions-constructs/
aws-lambda-dynamodb';
import * as lambda from 'aws-cdk-lib/aws-lambda';

const constructProps: LambdaToDynamoDBProps = {
  lambdaFunctionProps: {
    code: lambda.Code.fromAsset(`lambda`),
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler'
  },
};

new LambdaToDynamoDB(this, 'test-lambda-dynamodb-stack', constructProps);
```

Python

```
from aws_solutions_constructs.aws_lambda_dynamodb import LambdaToDynamoDBProps,
 LambdaToDynamoDB
from aws_cdk import (
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

LambdaToDynamoDB(self, 'test_lambda_dynamodb_stack',
                    lambda_function_props=_lambda.FunctionProps(
                        code=_lambda.Code.from_asset(
                            'lambda'),
                        runtime=_lambda.Runtime.PYTHON_3_9,
                        handler='index.handler'
                    ))
```

Java

```

import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.lambdadynamodb.*;

new LambdaToDynamoDB(this, "test_lambda_dynamodb_stack", new
 LambdaToDynamoDBProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
dynamoTableProps?	`dynamodb.TableProps`	Optional user provided props to override the default props for DynamoDB Table
existingTableObj?	`dynamodb.Table`	Existing instance of DynamoDB table object, providing both this and `dynamoTableProps` will cause an error.
tablePermissions?	`string`	Optional table permissions to grant to the Lambda function. One of the following may be specified: `All`, `Read`, `ReadWrite`, `Write`.
tableEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the name of the DynamoDB table. Default: DDB_TABLE_NAME
existingVpc?	`ec2.IVpc`	An optional, existing VPC into which this pattern should be deployed. When deployed in

Name	Type	Description
		a VPC, the Lambda function will use ENIs in the VPC to access network resources and a Gateway Endpoint will be created in the VPC for Amazon DynamoDB. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the [`ec2.Vpc.fromLookup()`]() method.
vpcProps?	[`ec2.VpcProps`]()	Optional user-provided properties to override the default properties for the new VPC. `enableDnsHostnames`, `enableDnsSupport`, `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of lambda.Function created by the construct
dynamoTable	`dynamodb.Table`	Returns an instance of dynamodb.Table created by the construct
vpc?	`ec2.IVpc`	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS Lambda Function

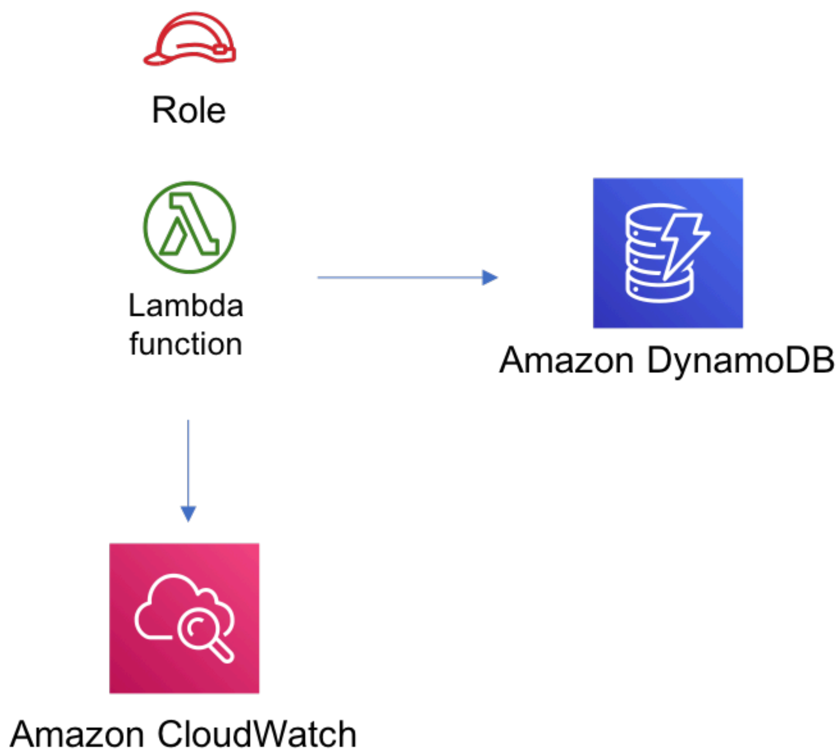
- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - (default) DDB_TABLE_NAME
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Amazon DynamoDB Table

- Set the billing mode for DynamoDB Table to On-Demand (Pay per request)
- Enable server-side encryption for DynamoDB Table using AWS managed KMS Key

- Creates a partition key called "id" for DynamoDB Table
- Retain the Table when deleting the CloudFormation stack
- Enable continuous backups and point-in-time recovery

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-lambda-dynamodb](https://github.com/awslabs/aws-solutions-constructs/tree/main/source/patterns/%40aws-solutions-constructs/aws-lambda-dynamodb)

aws-lambda-elasticachememcached

STABILITY EXPERIMENTAL

Language	Package
Python	aws_solutions_constructs.aws_lambda_elasticachememcached
Typescript	@aws-solutions-constructs/aws-lambda-elasticachememcached
Java	software.amazon.awsconstructs.services.lambdaelasticachememcached

Overview

This AWS Solutions Construct implements an AWS Lambda function connected to an Amazon Elasticache Memcached cluster.

Here is a minimal deployable pattern definition :

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { LambdaToElasticachememcached } from '@aws-solutions-constructs/aws-lambda-elasticachememcached';
import * as lambda from 'aws-cdk-lib/aws-lambda';

new LambdaToElasticachememcached(this, 'LambdaToElasticachememcachedPattern', {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_16_X,
        handler: 'index.handler',
        code: lambda.Code.fromAsset(`lambda`)
    }
});
```

Python

```
from aws_solutions_constructs.aws_lambda_elasticachememcached import
 LambdaToElasticachememcached
from aws_cdk import (
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

LambdaToElasticachememcached(self, 'LambdaToCachePattern',
        lambda_function_props=_lambda.FunctionProps(
            code=_lambda.Code.from_asset('lambda'),
            runtime=_lambda.Runtime.PYTHON_3_9,
            handler='index.handler'
        )
        )
```

Java

```

import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.lambdaelasticachememcached.*;

new LambdaToElasticachememcached(this, "LambdaToCachePattern", new
 LambdaToElasticachememcachedProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	Optional user provided props to override the default props for the Lambda function.
existingVpc?	`ec2.IVpc`	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to access network resources and an Interface Endpoint will be created in the VPC for Amazon Elasticache. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the `ec2.Vpc.fromLookup()` method.
vpcProps?	`ec2.VpcProps`	Optional user provided properties to override the default properties for the new VPC. `subnetConfiguration` is set by the pattern, so any values for those properties

Name	Type	Description
		supplied here will be overridden.
cacheEndpointEnvironmentVariableName?	string	Optional Name for the Lambda function environment variable set to the cache endpoint. Default: CACHE_ENDPOINT
cacheProps?	`cache.CfnCacheClusterProps`	Optional user provided props to override the default props for the Elasticache Cluster. Providing both this and `existingCache` will cause an error.
existingCache?	`cache.CfnCacheCluster`	Existing instance of Elasticache Cluster object, providing both this and `cacheProps` will cause an error. If you provide this, you must provide the associated VPC in existingVpc.

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of the Lambda function used by the pattern.
vpc	`ec2.IVpc`	Returns an interface on the VPC used by the pattern. This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

Name	Type	Description
cache	`cache.CfnCacheCluster`	The Elasticache Memcached cluster used by the construct.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

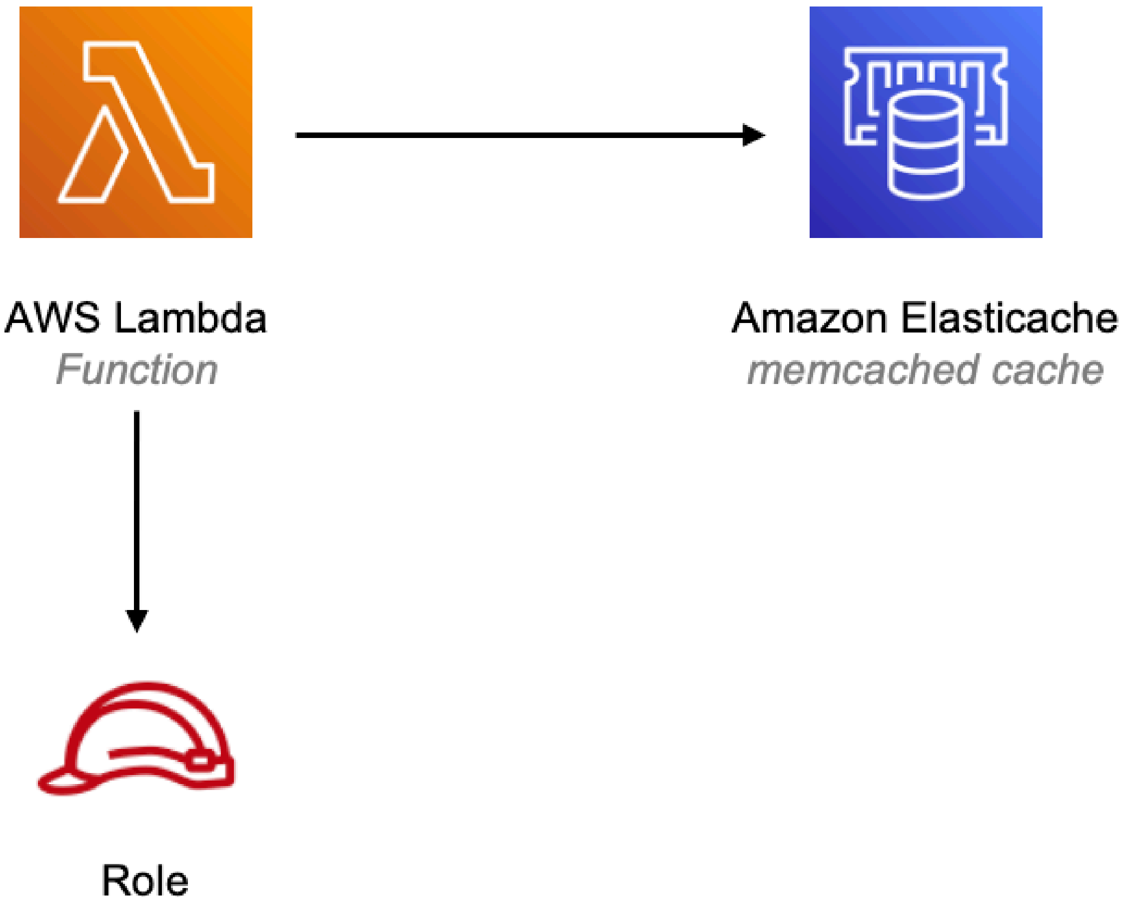
AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Attached to self referencing security group to grant access to cache
- Set Environment Variables
  - (default) CACHE_ENDPOINT
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Amazon Elasticache Memcached Cluster

- Creates multi node, cross-az cluster by default
  - 2 cache nodes, type: cache.t3.medium
- Self referencing security group attached to cluster endpoint

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-lambda-elasticachememcached

aws-lambda-elasticsearch-kibana

STABILITY EXPERIMENTAL

All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_lambda_elasticsearch_kibana
Typescript	@aws-solutions-constructs/aws-lambda-elasticsearch-kibana
Java	software.amazon.awsconstructs.services.lambdaelasticsearchkibana

Overview

This AWS Solutions Construct implements an AWS Lambda function and Amazon Elasticsearch Service with the least privileged permissions.

Some cluster configurations (e.g VPC access) require the existence of the `AWSServiceRoleForAmazonElasticsearchService` Service-Linked Role in your account.

You will need to create the service-linked role using the AWS CLI once in any account using this construct (it may have already been run to support other stacks):

```
aws iam create-service-linked-role --aws-service-name es.amazonaws.com
```

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps, Aws } from 'aws-cdk-lib';
```

```
import { LambdaToElasticSearchAndKibana } from '@aws-solutions-constructs/aws-lambda-elasticsearch-kibana';
import * as lambda from "aws-cdk-lib/aws-lambda";

const lambdaProps: lambda.FunctionProps = {
  code: lambda.Code.fromAsset(`lambda`),
  runtime: lambda.Runtime.NODEJS_16_X,
  handler: 'index.handler'
};

new LambdaToElasticSearchAndKibana(this, 'sample', {
  lambdaFunctionProps: lambdaProps,
  domainName: 'testdomain',
  // TODO: Ensure the Cognito domain name is globally unique
  cognitoDomainName: 'globallyuniquedomain' + Aws.ACCOUNT_ID
});
```

Python

```
from aws_solutions_constructs.aws_lambda_elasticsearch_kibana import
 LambdaToElasticSearchAndKibana
from aws_cdk import (
    aws_lambda as _lambda,
    Aws,
    Stack
)
from constructs import Construct

lambda_props = _lambda.FunctionProps(
    code=_lambda.Code.from_asset('lambda'),
    runtime=_lambda.Runtime.PYTHON_3_9,
    handler='index.handler'
)

LambdaToElasticSearchAndKibana(self, 'sample',
                            lambda_function_props=lambda_props,
                            domain_name='testdomain',
                            # TODO: Ensure the Cognito domain name is globally
 unique
                            cognito_domain_name='globallyuniquedomain' +
 Aws.ACCOUNT_ID
                            )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.Aws;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.lambdaelasticsearchkibana.*;

new LambdaToElasticSearchAndKibana(this, "sample",
        new LambdaToElasticSearchAndKibanaProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_16_X)
                        .code(Code.fromAsset("lambda"))
                        .handler("index.handler")
                        .build())
                .domainName("testdomain")
                // TODO: Ensure the Cognito domain name is globally unique
                .cognitoDomainName("globallyuniquedomain" + Aws.ACCOUNT_ID)
                .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.

Name	Type	Description
esDomainProps?	elasticsearch.CfnDomainProps	Optional user provided props to override the default props for the Elasticsearch Service
domainName	string	Domain name for the Cognito and the Elasticsearch Service
cognitoDomainName?	string	Optional Cognito Domain Name, if provided it will be used for Cognito Domain, and domainName will be used for the Elasticsearch Domain
createCloudWatchAlarms?	boolean	Whether to create recommended CloudWatch alarms
domainEndpointEnvironmentVariableName?	string	Optional Name for the ElasticSearch domain endpoint environment variable set for the Lambda function.
existingVpc?	ec2.IVpc	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to access network resources. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the `ec2.Vpc.fromLookup()` method.

Name	Type	Description
vpcProps?	`ec2.VpcProps`	Optional user provided properties to override the default properties for the new VPC. `enableDnsHostnames` , `enableDnsSupport` , `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of `lambda.Function` created by the construct
userPool	`cognito.UserPool`	Returns an instance of `cognito.UserPool` created by the construct

Name	Type	Description
userPoolClient	cognito.UserPoolClient	Returns an instance of `cognito.UserPoolClient` created by the construct
identityPool	cognito.CfnIdentityPool	Returns an instance of `cognito.CfnIdentityPool` created by the construct
elasticsearchDomain	elasticsearch.CfnDomain	Returns an instance of `elasticsearch.CfnDomain` created by the construct
elasticsearchRole	iam.Role	Returns an instance of `iam.Role` created by the construct for `elasticsearch.CfnDomain`
cloudwatchAlarms?	cloudwatch.Alarm[]	Returns a list of `cloudwatch.Alarm` created by the construct
vpc?	ec2.IVpc	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

Lambda Function

This pattern requires a lambda function that can post data into the Elasticsearch. A sample function is provided here.

Default settings

Out of the box implementation of the Construct without any overrides will set the following defaults:

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for Node.js Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - (default) DOMAIN_ENDPOINT
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED

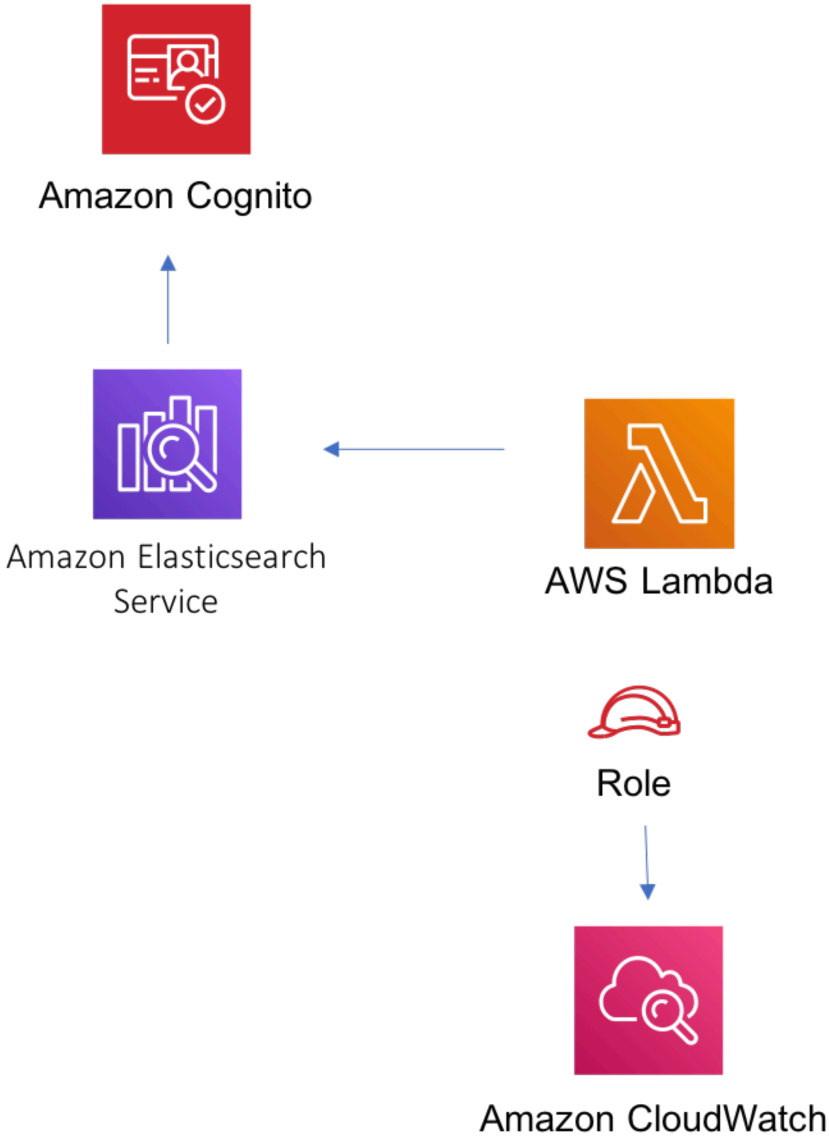
Amazon Cognito

- Set password policy for User Pools
- Enforce the advanced security mode for User Pools

Amazon Elasticsearch Service

- Deploy best practices CloudWatch Alarms for the Elasticsearch Service domain
- Secure the Kibana dashboard access with Cognito User Pools
- Enable server-side encryption for the Elasticsearch Service domain using AWS managed KMS Key
- Enable node-to-node encryption for the Elasticsearch Service domain
- Configure the cluster for the Elasticsearch Service domain

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-lambda-elasticsearch-kibana

aws-lambda-eventbridge

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_lambda_eventbridge
Typescript	@aws-solutions-constructs/aws-lambda-eventbridge
Java	software.amazon.awsconstructs.services.lambdaeventbridge

Overview

This AWS Solutions Construct implements an AWS Lambda function connected to an Amazon EventBridge.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps, Aws } from 'aws-cdk-lib';
import { LambdaToEventbridge, LambdaToEventbridgeProps } from "@aws-solutions-constructs/aws-lambda-eventbridge";
import * as lambda from 'aws-cdk-lib/aws-lambda';

new LambdaToEventbridge(this, 'LambdaToEventbridgePattern', {
  lambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_16_X,
```

```
    handler: 'index.handler',
    code: lambda.Code.fromAsset(`lambda`)
  }
});
```

Python

```
from aws_solutions_constructs.aws_lambda_eventbridge import LambdaToEventbridge
from aws_cdk import (
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

LambdaToEventbridge(self, 'LambdaToEventbridgePattern',
                    lambda_function_props=_lambda.FunctionProps(
                        code=_lambda.Code.from_asset('lambda'),
                        runtime=_lambda.Runtime.PYTHON_3_9,
                        handler='index.handler'
                    )
                    )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.lambdaeventbridge.*;

new LambdaToEventbridge(this, "LambdaToEventbridgePattern", new
 LambdaToEventbridgeProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	An optional, existing Lambda function to be used instead of the default function. Providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	Optional user-provided properties to override the default properties for the Lambda function.
existingEventBusInterface?	`events.IEventBus`	Optional user-provided custom EventBus for construct to use. Providing both this and `eventBusProps` results an error.
eventBusProps?	`events.EventBusProps`	Optional user-provided properties to override the default properties when creating a custom EventBus. Setting this value to `{}` will create a custom EventBus using all default properties. If neither this nor `existingEventBusInterface` is provided the construct will use the `default` EventBus. Providing both this and `existingEventBusInterface` results an error.

Name	Type	Description
existingVpc?	[ec2.IVpc](#)	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to access network resources and an Interface Endpoint will be created in the VPC for Amazon EventBridge. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the [ec2.Vpc.fromLookup()](#) method.
vpcProps?	[ec2.VpcProps](#)	Optional user-provided properties to override the default properties for the new VPC. `enableDnsHostnames`, `enableDnsSupport`, `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.

Name	Type	Description
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:
eventBusEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the name of the Event bus. Default: EVENTBUS_NAME

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of the Lambda function created by the pattern.
eventBus?	`events.IEventBus`	Returns the instance of events.IEventBus used by the construct
vpc?	`ec2.IVpc`	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

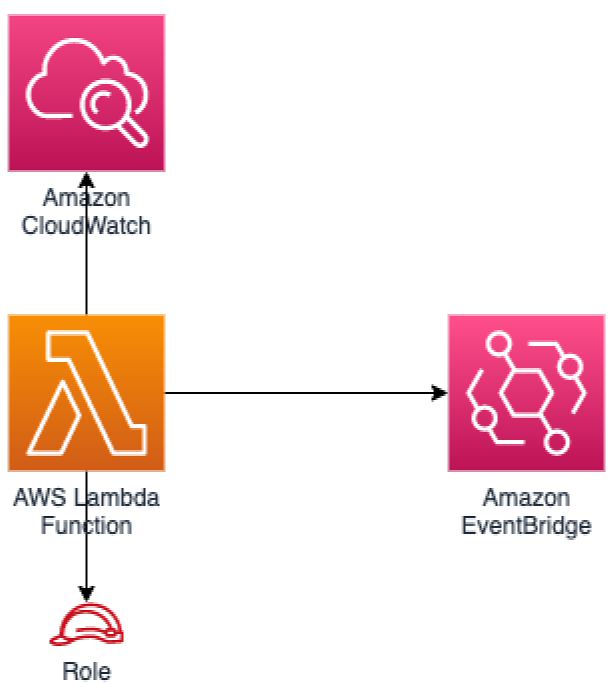
Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function.
- Enable reusing connections with Keep-Alive for NodeJs Lambda function.
- Allow the function to put events to EventBus (custom EventBus can be used by specifying `existingEventBusInterface` or `eventBusProps` property).
- Enable X-Ray Tracing
- Set Environment Variables
  - EVENTBUS_NAME
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-lambda-eventbridge]()

aws-lambda-kendra

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_lambda_kendra
Typescript	@aws-solutions-constructs/aws-lambda-kendra
Java	software.amazon.awsconstructs.services.lambdakendra

Overview

This AWS Solutions Construct implements an AWS Lambda function and Amazon Kendra index with the least privileged permissions.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps, Aws } from 'aws-cdk-lib';
import { LambdaToKendra } from '@aws-solutions-constructs/aws-lambda-kendra';
import * as lambda from "aws-cdk-lib/aws-lambda";
import * as s3 from "aws-cdk-lib/aws-s3";

const lambdaProps: lambda.FunctionProps = {
  code: lambda.Code.fromAsset(`lambda`),
  runtime: lambda.Runtime.NODEJS_18_X,
  handler: 'index.handler'
```

```
};

new LambdaToKendra(this, 'sample', {
  lambdaFunctionProps: lambdaProps,
  kendraIndexProps: {},
  kendraDataSourceProps: [{
    type: 'S3',
    dataSourceConfiguration: {
      s3Configuration: {
        bucketName: 'your-bucket-name',
      }
    }
  ],
});
```

Python

```
TBD
```

Java

```
TBD
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.

Name	Type	Description
kendraIndexProps?	[kendra.CfnIndexProps](#)	Optional user provided props to override the default props for the Kendra index. Providing both these and existingKendraIndexObj is an error.
kendraDataSourcesProps	[CfnDataSourceProps[]](#)	A list of data sources that will provide data to the Kendra index. *At least 1 must be specified*. We will do majority of processing for some data sources (S3 crawler initially), but for others the props must be complete (e.g. proper roleArn, etc.)
indexPermissions?	`string[]`	Optional - index permissions to grant to the Lambda function. One or more of the following may be specified: `Read`, `SubmitFeedback` and `Write`. Default is `["Read", "SubmitFeedback"]` . Read is all the operations IAM defines as Read and List. SubmitFeedback is only the SubmitFeedback action. Write is all the operations IAM defines as Write and Tag. This functionality may be overridden by providing a specific role arn in lambdaFunctionProps

Name	Type	Description
existingKendraIndexObj?	`kendra.cfnIndex`	An existing Kendra index to which the Lambda function will be granted access. Supplying along with kendraIndexProps or kendraDataSourceProps will throw an error.
existingVpc?	`ec2.IVpc`	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to access network resources. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the `ec2.Vpc.fromLookup()` method.

Name	Type	Description
vpcProps?	ec2.VpcProps	Optional user provided properties to override the default properties for the new VPC. `enableDnsHostnames` , `enableDnsSupport` , `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to `true` will deploy the minimal, most private VPC to run the pattern, consisting of (1) one isolated subnet in each Availability Zone used by the CDK program; and (2) `enableDnsHostnames` and `enableDnsSupport` both being set to `true`. If this property is `true` then `existingVpc` cannot be specified. Defaults to `false`.

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of `lambda.Function` managed by the construct
kendraIndex	`kendra.cfnIndex`	Returns an instance of `kendra.cfnIndex` managed by the construct
kendraDataSources	DataSourceProperties[] (this interface is defined by Solutions Constructs and described below)	A list of data sources created for this construct/index, each in an object that includes the role for that data source.
lambdaRole	`iam.Role`	The role assumed by the Lambda function
vpc?	`ec2.IVpc`	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

interface DataSourceProperties { role?: `iam.Role`, source: | `CfnDataSource` } ## Lambda Function

This pattern requires a lambda function that can access a Kendra index.

Default settings

Out of the box implementation of the Construct without any overrides will set the following defaults:

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for Node.js Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - (default) KENDRA_INDEX_ID
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED

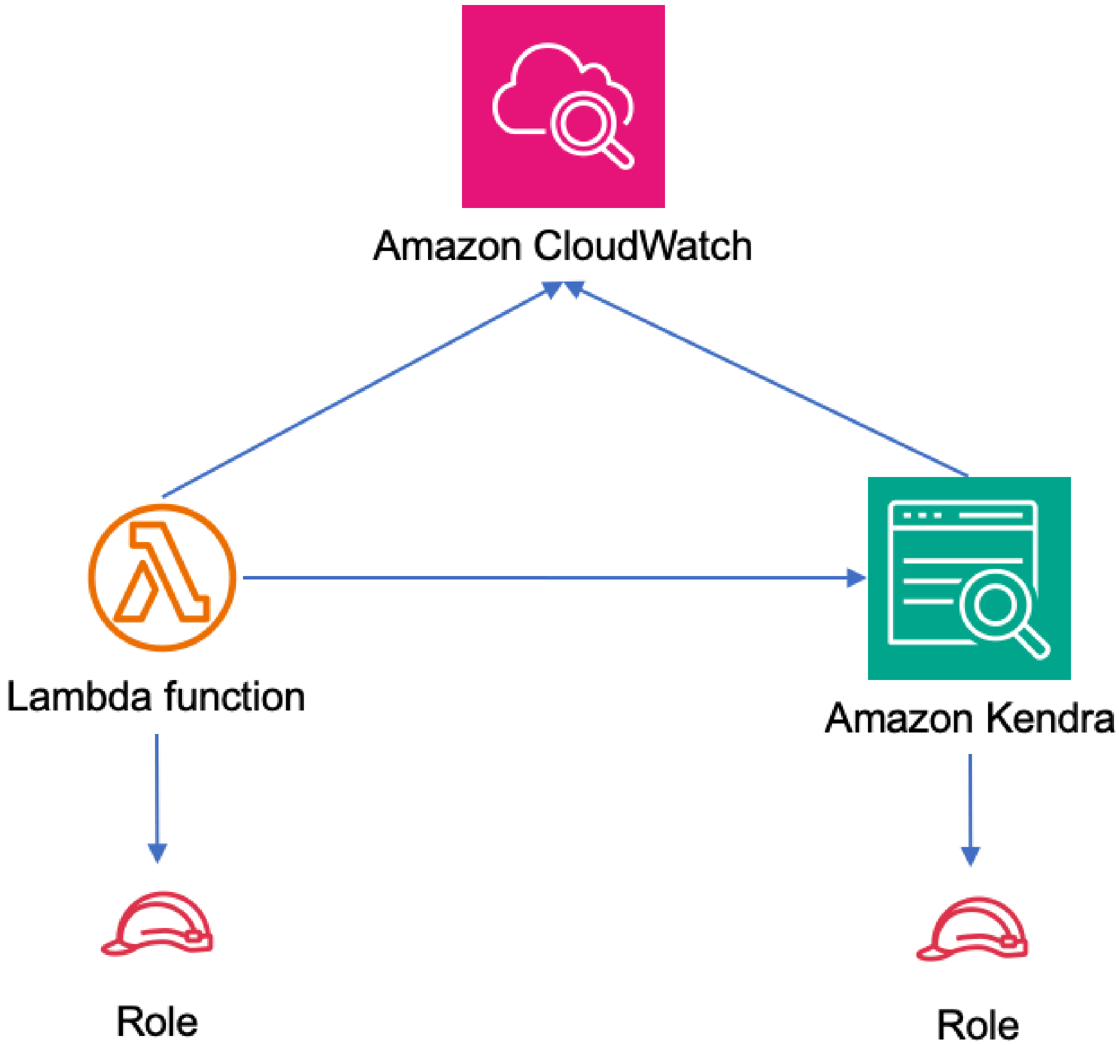
Amazon Kendra Index

- Creates Amazon Kendra endpoint in VPC if appropriate
- Defaults to DEVELOPER_EDITION

Amazon Kendra DataSources

- Sets up correct IAM roles to access data for:
  - S3 data sources
  - Which others should we support in MLP? https://docs.aws.amazon.com/kendra/latest/dg/iam-roles.html
- Adds each data source to Kendra index

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:

	@aws-solutions-constructs/aws-lambda-kendra

aws-lambda-kinesisfirehose

STABILITY EXPERIMENTAL

Language	Package
Python	aws_solutions_constructs.aws_lambda_kinesisfirehose
Typescript	@aws-solutions-constructs/aws-lambda-kinesisfirehose
Java	software.amazon.awsconstructs.services.lambdakinesisfirehose

Overview

This AWS Solutions Construct implements an AWS Lambda function connected to an existing Amazon Kinesis Firehose Delivery Stream.

Here is a minimal deployable pattern definition :

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { LambdaToS3 } from '@aws-solutions-constructs/aws-lambda-kinesisfirehose';
import * as lambda from 'aws-cdk-lib/aws-lambda';

// The construct requires an existing Firehose Delivery Stream, this can be created
 in raw CDK or extracted
// from a previously instantiated construct that created an Firehose Delivery Stream
const existingFirehoseDeliveryStream =
 previouslyCreatedKinesisFirehoseToS3Construct.kinesisFirehose;

new LambdaToKinesisFirehose(this, 'LambdaToFirehosePattern', {
  lambdaFunctionProps: {
      runtime: lambda.Runtime.NODEJS_16_X,
      handler: 'index.handler',
```

```
      code: lambda.Code.fromAsset(`lambda`)
  },
  existingKinesisFirehose: existingFirehoseDeliveryStream
});
```

Python

```
from aws_solutions_constructs.aws_lambda_kinesisfirehose import
 LambdaToKinesisFirehose
from aws_cdk import (
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

# The construct requires an existing Firehose Delivery Stream, this can be created
 in raw CDK or extracted
# from a previously instantiated construct that created an Firehose Delivery Stream
existingFirehoseDeliveryStream =
 previouslyCreatedKinesisFirehoseToS3Construct.kinesisFirehose;

LambdaToKinesisFirehose(self, 'LambdaToFirehosePattern',
        existingKinesisFirehose=existingFirehoseDeliveryStream,
        lambda_function_props=_lambda.FunctionProps(
            code=_lambda.Code.from_asset('lambda'),
            runtime=_lambda.Runtime.PYTHON_3_9,
            handler='index.handler'
        )
        )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.lambdakinesisfirehose.*;
```

```
// The construct requires an existing Firehose Delivery Stream, this can be created
 in raw CDK or extracted
// from a previously instantiated construct that created an Firehose Delivery Stream
existingFirehoseDeliveryStream =
 previouslyCreatedKinesisFirehoseToS3Construct.kinesisFirehose;

new LambdaToKinesisFirehose(this, "LambdaToFirehosePattern", new
 LambdaToKinesisFirehoseProps.Builder()
        .existingKinesisFirehose(existingFirehoseDeliveryStream)
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	Optional user provided props to override the default props for the Lambda function.
existingKinesisFirehose	kinesisfirehose.CfnDeliveryStream	An existing Kinesis Firehose Delivery Stream to which the Lambda function can put data. Note - the delivery stream construct must have already been created and have the deliveryStreamName set. This

Name	Type	Description
		construct will *not* create a new Delivery Stream.
existingVpc?	`ec2.IVpc`	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to access network resources and an Interface Endpoint will be created in the VPC for Amazon Kinesis Data Firehose. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the `ec2.Vpc.fromLookup()` method.
vpcProps?	`ec2.VpcProps`	Optional user provided properties to override the default properties for the new VPC. `enableDnsHostnames`, `enableDnsSupport`, `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.

Name	Type	Description
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:
firehoseEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the name of the delivery stream. Default: FIREHOSE_DELIVERYSTREAM_NAME

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of the Lambda function created by the pattern.
kinesisFirehose	kinesisfirehose.CfnDeliveryStream	The Kinesis Firehose Delivery Stream used by the construct.
vpc?	`ec2.IVpc`	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

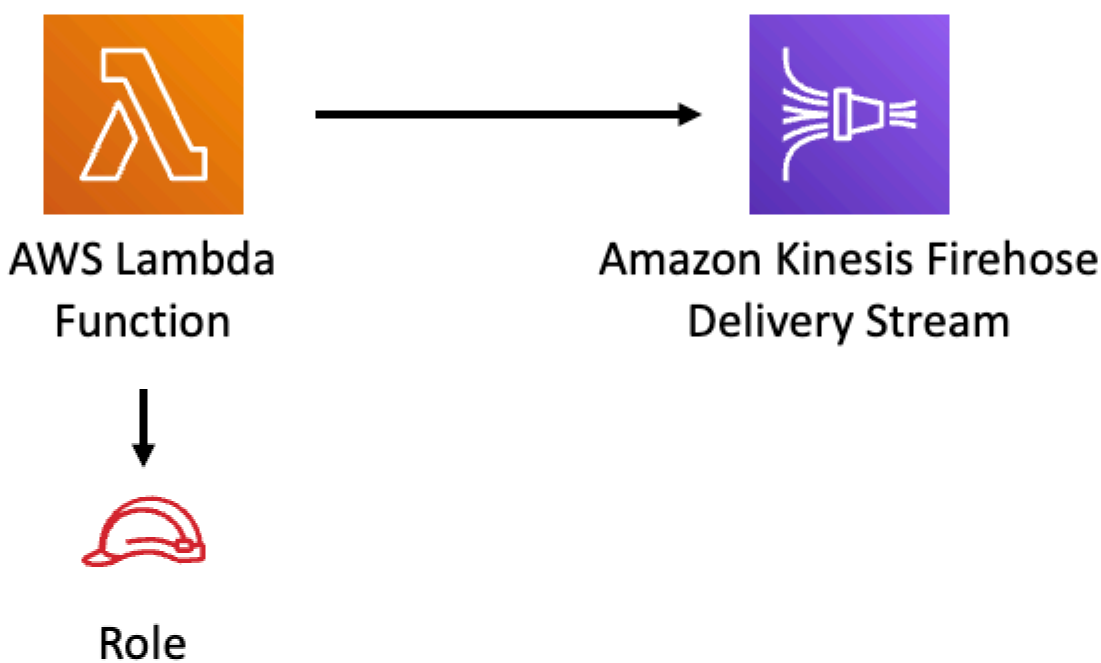
AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - (default) FIREHOSE_DELIVERYSTREAM_NAME
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED

Amazon Kinesis Firehose Delivery Stream

- This construct must be provided a configured Stream construct, it does not change this Stream.

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-lambda-kinesisfirehose](#)

aws-lambda-kinesisstreams

STABILITY EXPERIMENTAL

Language	Package
Python	`aws_solutions_constructs.aws_lambda_kinesis_stream`
Typescript	`@aws-solutions-constructs/aws-lambda-kinesisstreams`
Java	`software.amazon.awsconstructs.services.lambdakinesisstreams`

Overview

This AWS Solutions Construct deploys an AWS Lambda Function that can put records on an Amazon Kinesis Data Stream.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { LambdaToKinesisStreamsProps } from '@aws-solutions-constructs/aws-lambda-kinesisstreams';
import * as lambda from 'aws-cdk-lib/aws-lambda';
```

```
new LambdaToKinesisStreams(this, 'LambdaToKinesisStreams', {
  lambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_18_X,
    handler: 'index.handler',
    code: lambda.Code.fromAsset(`lambda`)
  }
});
```

Python

```
from aws_solutions_constructs.aws_lambda_kinesis_stream import
 LambdaToKinesisStreams
from aws_cdk import (
    aws_lambda as _lambda,
    aws_kinesis as kinesis,
    Stack
)
from constructs import Construct

LambdaToKinesisStreams(self, 'LambdaToKinesisStreams',
                        lambda_function_props=_lambda.FunctionProps(
                          runtime=_lambda.Runtime.PYTHON_3_9,
                          handler='index.handler',
                          code=_lambda.Code.from_asset('lambda')
                        )
                      )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.eventsources.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.lambdakinesisstreams.*;

new LambdaToKinesisStreams(this, "LambdaToKinesisStreams", new
 LambdaToKinesisStreamsProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
```

```
                .runtime(Runtime.NODEJS_18_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of a Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda Function.
existingStreamObj?	`kinesis.Stream`	Existing instance of a Kinesis Data Stream, providing both this and `kinesisStreamProps` will cause an error.
kinesisStreamProps?	`kinesis.StreamProps`	Optional user-provided props to override the default props for the Kinesis Data Stream.
createCloudWatchAlarms	`boolean`	Whether to create recommended CloudWatch Alarms (defaults to true).
existingVpc?	`ec2.IVpc`	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function

Name	Type	Description
		will use ENIs in the VPC to access network resources and an Interface Endpoint will be created in the VPC for Amazon Kinesis Streams. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the [`ec2.Vpc.fromLookup()`](#) method.
vpcProps?	[`ec2.VpcProps`](#)	Optional user-provided properties to override the default properties for the new VPC. `enableDnsHostnames`, `enableDnsSupport`, `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:

Name	Type	Description
streamEnvironmentVariableName?	`string`	Optional Name to override the Lambda Function default environment variable name that holds the Kinesis Data Stream name value. Default: KINESIS_DATASTREAM_NAME

Pattern Properties

Name	Type	Description
lambdaFunction	[`lambda.Function`]()	Returns an instance of the Lambda Function.
kinesisStream	[`kinesis.Stream`]()	Returns an instance of the Kinesis Data Stream.
cloudwatchAlarms?	[`cloudwatch.Alarm[]`]()	Returns the CloudWatch Alarms created to monitor the Kinesis Data Stream.
vpc?	[`ec2.IVpc`]()	Returns an interface to the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

Default settings

Out of the box implementation of the Construct without any overrides will set the following defaults:

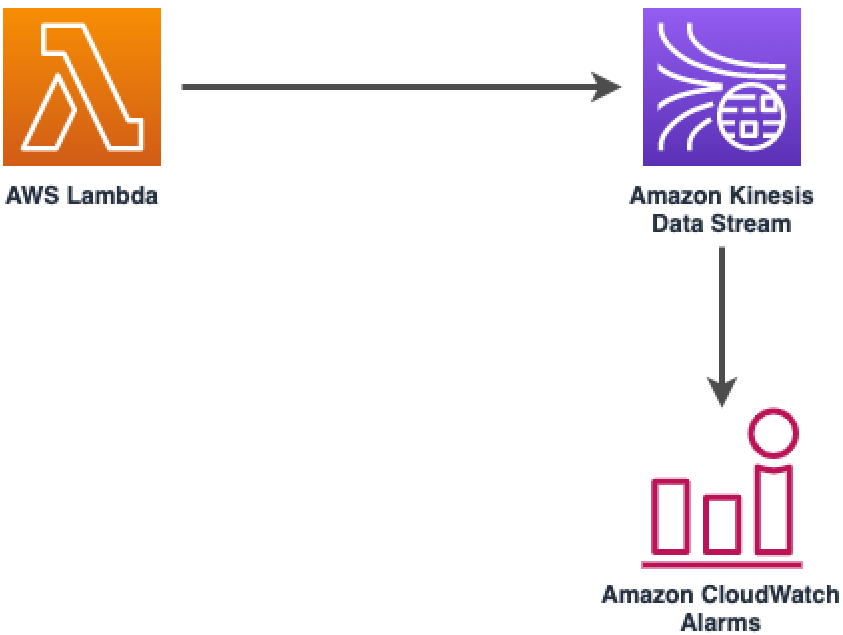
AWS Lambda Function

- Minimally-permissive IAM role for the Lambda Function to put records on the Kinesis Data Stream
- Enable X-Ray Tracing
- Sets an Environment Variable named KINESIS_DATASTREAM_NAME that holds the Kinesis Data Stream Name, which is a required property Kinesis Data Streams SDK when making calls to it

Amazon Kinesis Stream

- Enable server-side encryption for the Kinesis Data Stream using AWS Managed CMK
- Deploy best practices CloudWatch Alarms for the Kinesis Data Stream

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:

@aws-solutions-constructs/aws-lambda-kinesisstreams

aws-lambda-opensearch

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_lambda_opensearch
Typescript	@aws-solutions-constructs/aws-lambda-opensearch
Java	software.amazon.awsconstructs.services.lambdaopensearch

Overview

This AWS Solutions Construct implements an AWS Lambda function and Amazon OpenSearch Service with the least privileged permissions.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps, Aws } from 'aws-cdk-lib';
import { LambdaToOpenSearch } from '@aws-solutions-constructs/aws-lambda-
opensearch';
import * as lambda from "aws-cdk-lib/aws-lambda";

const lambdaProps: lambda.FunctionProps = {
  code: lambda.Code.fromAsset(`lambda`),
  runtime: lambda.Runtime.NODEJS_16_X,
  handler: 'index.handler'
```

```
};

new LambdaToOpenSearch(this, 'sample', {
  lambdaFunctionProps: lambdaProps,
  openSearchDomainName: 'testdomain',
  // TODO: Ensure the Cognito domain name is globally unique
  cognitoDomainName: 'globallyuniquedomain' + Aws.ACCOUNT_ID
});
```

Python

```
from aws_solutions_constructs.aws_lambda_opensearch import LambdaToOpenSearch
from aws_cdk import (
    aws_lambda as _lambda,
    Aws,
    Stack
)
from constructs import Construct

lambda_props = _lambda.FunctionProps(
    code=_lambda.Code.from_asset('lambda'),
    runtime=_lambda.Runtime.PYTHON_3_9,
    handler='index.handler'
)

LambdaToOpenSearch(self, 'sample',
                            lambda_function_props=lambda_props,
                            open_search_domain_name='testdomain',
                            # TODO: Ensure the Cognito domain name is globally
 unique
                            cognito_domain_name='globallyuniquedomain' +
 Aws.ACCOUNT_ID
                            )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.Aws;
```

```
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.lambdaopensearch.*;

new LambdaToOpenSearch(this, "sample",
        new LambdaToOpenSearchProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_16_X)
                        .code(Code.fromAsset("lambda"))
                        .handler("index.handler")
                        .build())
                .openSearchDomainName("testdomain")
                // TODO: Ensure the Cognito domain name is globally unique
                .cognitoDomainName("globallyuniquedomain" + Aws.ACCOUNT_ID)
                .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
openSearchDomainProps?	`opensearchservice.CfnDomainProps`	Optional user provided props to override the default props for the OpenSearch Service.
openSearchDomainName	`string`	Domain name for the OpenSearch Service.
cognitoDomainName?	`string`	Optional Amazon Cognito domain name. If omitted the

Name	Type	Description
		Amazon Cognito domain will default to the OpenSearch Service domain name.
createCloudWatchAlarms?	`boolean`	Whether to create the recommended CloudWatch alarms.
domainEndpointEnvironmentVariableName?	`string`	Optional name for the OpenSearch domain endpoint environment variable set for the Lambda function. Default is `DOMAIN_ENDPOINT` .
existingVpc?	[`ec2.IVpc`]()	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to access network resources. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the [`ec2.Vpc.fromLookup()`]() method.

Name	Type	Description
vpcProps?	`ec2.VpcProps`	Optional user provided properties to override the default properties for the new VPC. `enableDnsHostnames` , `enableDnsSupport` , `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of `lambda.Function` created by the construct
userPool	`cognito.UserPool`	Returns an instance of `cognito.UserPool` created by the construct

Name	Type	Description
userPoolClient	`cognito.UserPoolClient`	Returns an instance of `cognito.UserPoolClient` created by the construct
identityPool	`cognito.CfnIdentityPool`	Returns an instance of `cognito.CfnIdentityPool` created by the construct
openSearchDomain	`opensearchservice.CfnDomain`	Returns an instance of `opensearch.CfnDomain` created by the construct
openSearchRole	`iam.Role`	Returns an instance of `iam.Role` created by the construct for `opensearch.CfnDomain`
cloudWatchAlarms?	`cloudwatch.Alarm[]`	Returns a list of `cloudwatch.Alarm` created by the construct
vpc?	`ec2.IVpc`	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

Lambda Function

This pattern requires a lambda function that can post data into the OpenSearch. A sample function is provided here.

Default settings

Out of the box implementation of the Construct without any overrides will set the following defaults:

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for Node.js Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - (default) DOMAIN_ENDPOINT
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED

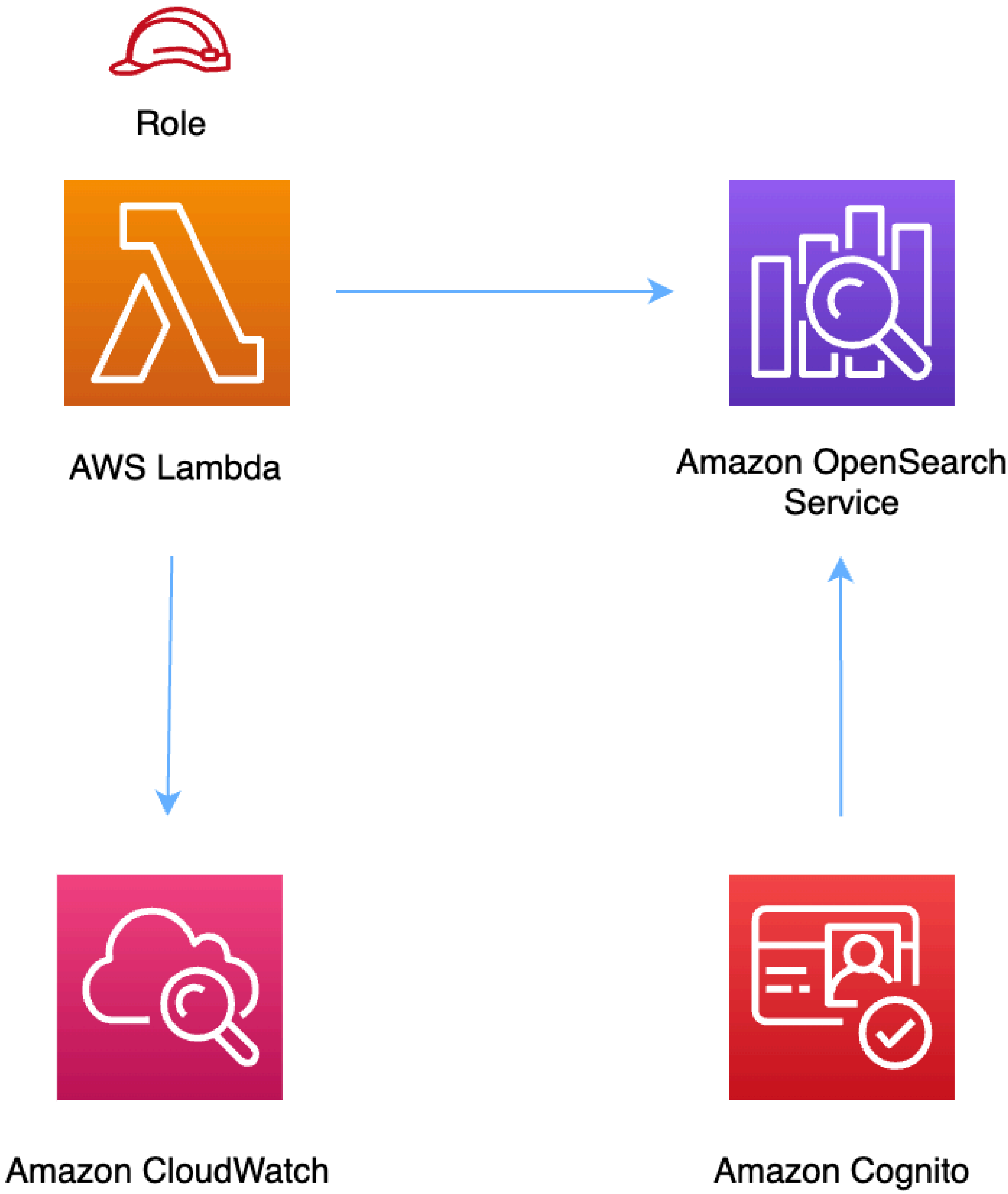
Amazon Cognito

- Set password policy for User Pools
- Enforce the advanced security mode for User Pools

Amazon OpenSearch Service

- Deploy best practices CloudWatch Alarms for the OpenSearch Service domain
- Secure the OpenSearch Service dashboard access with Cognito User Pools
- Enable server-side encryption for OpenSearch Service domain using AWS managed KMS Key
- Enable node-to-node encryption for the OpenSearch Service domain
- Configure the cluster for the OpenSearch Service domain

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-lambda-opensearch

aws-lambda-s3

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_lambda_s3`
Typescript	`@aws-solutions-constructs/aws-lambda-s3`
Java	`software.amazon.awsconstructs.services.lambdas3`

Overview

This AWS Solutions Construct implements an AWS Lambda function connected to an Amazon S3 bucket.

Here is a minimal deployable pattern definition :

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { LambdaToS3 } from '@aws-solutions-constructs/aws-lambda-s3';
import * as lambda from 'aws-cdk-lib/aws-lambda';

new LambdaToS3(this, 'LambdaToS3Pattern', {
```

```
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_16_X,
        handler: 'index.handler',
        code: lambda.Code.fromAsset(`lambda`)
    }
});
```

Python

```
from aws_solutions_constructs.aws_lambda_s3 import LambdaToS3
from aws_cdk import (
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

LambdaToS3(self, 'LambdaToS3Pattern',
        lambda_function_props=_lambda.FunctionProps(
            code=_lambda.Code.from_asset('lambda'),
            runtime=_lambda.Runtime.PYTHON_3_9,
            handler='index.handler'
        )
        )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.lambdas3.*;

new LambdaToS3(this, "LambdaToS3Pattern", new LambdaToS3Props.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	lambda.Function	Existing instance of Lambda Function object, providing both this and lambdaFunctionProps will cause an error.
lambdaFunctionProps?	lambda.FunctionProps	Optional user provided props to override the default props for the Lambda function.
existingBucketObj?	s3.IBucket	Existing instance of S3 Bucket object. If this is provided, then also providing bucketProps is an error.
bucketProps?	s3.BucketProps	Optional user provided props to override the default props for the S3 Bucket.
bucketPermissions?	string[]	Optional bucket permissions to grant to the Lambda function. One or more of the following may be specified: Delete, Put, Read, ReadWrite, Write.
existingVpc?	ec2.IVpc	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to access network resources and an Interface Endpoint will be created in the VPC for Amazon S3. If an existing VPC

Name	Type	Description
		is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the [`ec2.Vpc.fromLookup()`](#) method.
vpcProps?	[`ec2.VpcProps`](#)	Optional user provided properties to override the default properties for the new VPC. `enableDnsHostnames` , `enableDnsSupport` , `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:
bucketEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the name of the bucket. Default: S3_BUCKET_NAME

Name	Type	Description
loggingBucketProps?	[s3.BucketProps](#)	Optional user provided props to override the default props for the S3 Logging Bucket.
logS3AccessLogs?	boolean	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with associated storage costs for the logs. Enabling Access Logging is a best practice. default - true

Pattern Properties

Name	Type	Description
lambdaFunction	[lambda.Function](#)	Returns an instance of the Lambda function created by the pattern.
s3Bucket?	[s3.Bucket](#)	Returns an instance of the S3 bucket created by the pattern.
s3LoggingBucket?	[s3.Bucket](#)	Returns an instance of s3.Bucket created by the construct as the logging bucket for the primary bucket.
vpc?	[ec2.IVpc](#)	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.
s3BucketInterface	[s3.IBucket](#)	Returns an instance of s3.IBucke t created by the construct.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - (default) S3_BUCKET_NAME
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Amazon S3 Bucket

- Configure Access logging for S3 Bucket
- Enable server-side encryption for S3 Bucket using AWS managed KMS Key
- Enforce encryption of data in transit
- Turn on the versioning for S3 Bucket
- Don't allow public access for S3 Bucket
- Retain the S3 Bucket when deleting the CloudFormation stack
- Applies Lifecycle rule to move noncurrent object versions to Glacier storage after 90 days

Architecture

GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-lambda-s3

aws-lambda-sagemakerendpoint

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_lambda_sagemakerendpoint`
Typescript	`@aws-solutions-constructs/aws-lambda-sagemakerendpoint`
Java	`software.amazon.awsconstructs.services.lambdasagemakerendpoint`

Overview

This AWS Solutions Construct implements an AWS Lambda function connected to an Amazon Sagemaker Endpoint.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps, Duration } from 'aws-cdk-lib';
import * as lambda from 'aws-cdk-lib/aws-lambda';
import { LambdaToSagemakerEndpoint, LambdaToSagemakerEndpointProps } from '@aws-
solutions-constructs/aws-lambda-sagemakerendpoint';

const constructProps: LambdaToSagemakerEndpointProps = {
  modelProps: {
    primaryContainer: {
      image: '<AccountId>.dkr.ecr.<region>.amazonaws.com/linear-learner:latest',
      modelDataUrl: "s3://<bucket-name>/<prefix>/model.tar.gz",
    },
  },
  lambdaFunctionProps: {
    runtime: lambda.Runtime.PYTHON_3_8,
    code: lambda.Code.fromAsset(`lambda`),
    handler: 'index.handler',
    timeout: Duration.minutes(5),
    memorySize: 128,
  },
};

new LambdaToSagemakerEndpoint(this, 'LambdaToSagemakerEndpointPattern',
 constructProps);
```

Python

```
from constructs import Construct
from aws_solutions_constructs.aws_lambda_sagemakerendpoint import
 LambdaToSagemakerEndpoint, LambdaToSagemakerEndpointProps
from aws_cdk import (
    aws_lambda as _lambda,
    aws_sagemaker as sagemaker,
    Duration,
    Stack
)
from constructs import Construct

LambdaToSagemakerEndpoint(
```

```
    self, 'LambdaToSagemakerEndpointPattern',
    model_props=sagemaker.CfnModelProps(
        primary_container=sagemaker.CfnModel.ContainerDefinitionProperty(
            image='<AccountId>.dkr.ecr.<region>.amazonaws.com/linear-
learner:latest',
            model_data_url='s3://<bucket-name>/<prefix>/model.tar.gz',
        ),
        execution_role_arn="executionRoleArn"
    ),
    lambda_function_props=_lambda.FunctionProps(
        code=_lambda.Code.from_asset('lambda'),
        runtime=_lambda.Runtime.PYTHON_3_9,
        handler='index.handler',
        timeout=Duration.minutes(5),
        memory_size=128
    ))
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.Duration;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awscdk.services.sagemaker.*;
import software.amazon.awsconstructs.services.lambdasagemakerendpoint.*;

new LambdaToSagemakerEndpoint(this, "LambdaToSagemakerEndpointPattern",
        new LambdaToSagemakerEndpointProps.Builder()
                .modelProps(new CfnModelProps.Builder()
                        .primaryContainer(new
 CfnModel.ContainerDefinitionProperty.Builder()
                                .image("<AccountId>.dkr.ecr.<region>.amazonaws.com/
linear_learner:latest")
                                .modelDataUrl("s3://<bucket_name>/<prefix>/
model.tar.gz")
                                .build())
                        .executionRoleArn("executionRoleArn")
                        .build())
                .lambdaFunctionProps(new FunctionProps.Builder()
```

```
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .timeout(Duration.minutes(5))
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	[lambda.Function](#)	An optional, existing Lambda function to be used instead of the default function. Providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	[lambda.FunctionProps](#)	Optional user-provided properties to override the default properties for the Lambda function.
existingSagemakerEndpointObj?	[sagemaker.CfnEndpoint](#)	An optional, existing SageMaker Endpoint to be used. Providing both this and `endpointProps?` will cause an error.
modelProps?	[sagemaker.CfnModelProps](#) | any	User-provided properties to override the default properties for the SageMaker Model. At least `modelProps?.primaryContainer` must be provided to create a model. By default, the pattern will create a role with the minimum required permissions, but the

Name	Type	Description
		client can provide a custom role with additional capabilities using `modelProps?.executionRoleArn` .
endpointConfigProps?	[sagemaker.CfnEndpointConfigProps]()	Optional user-provided properties to override the default properties for the SageMaker Endpoint Config.
endpointProps?	[sagemaker.CfnEndpointProps]()	Optional user-provided properties to override the default properties for the SageMaker Endpoint Config.
existingVpc?	[ec2.IVpc]()	An optional, existing VPC into which this construct should be deployed. When deployed in a VPC, the Lambda function and Sagemaker Endpoint will use ENIs in the VPC to access network resources. An Interface Endpoint will be created in the VPC for Amazon SageMaker Runtime, and Amazon S3 VPC Endpoint. If an existing VPC is provided, the `deployVpc?` property cannot be `true`.

Name	Type	Description
vpcProps?	[ec2.VpcProps](#)	Optional user-provided properties to override the default properties for the new VPC. `enableDnsHostnames` , `enableDnsSupport` , `natGateways` and `subnetConfiguration` are set by the Construct, so any values for those properties supplied here will be overridden. If `deployVpc?` is not `true` then this property will be ignored.
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:
sagemakerEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the name of the SageMaker endpoint. Default: SAGEMAKER_ENDPOINT_NAME

Pattern Properties

Name	Type	Description
lambdaFunction	[lambda.Function](#)	Returns an instance of the Lambda function created by the pattern.

Name	Type	Description
sagem Endpoi	sagemaker.CfnEndpoint	Returns an instance of the SageMaker Endpoint created by the pattern.
sagem Endpoi onfig?	sagemaker.CfnEndpointConfig	Returns an instance of the SageMaker EndpointConfig created by the pattern, if `existingSagemakerEndpointObj?` is not provided.
sagem Model?	sagemaker.CfnModel	Returns an instance of the SageMaker Model created by the pattern, if `existingSagemakerEndpointObj?` is not provided.
vpc?	`ec2.IVpc`	Returns an instance of the VPC created by the pattern, if `deployVpc?` is `true`, or `existingVpc?` is provided.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

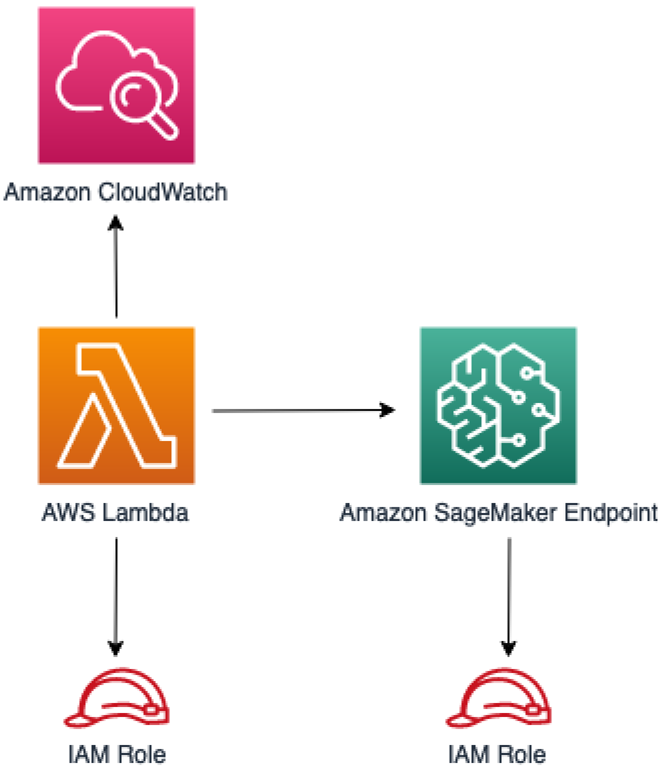
AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Allow the function to invoke the SageMaker endpoint for Inferences
- Configure the function to access resources in the VPC, where the SageMaker endpoint is deployed
- Enable X-Ray Tracing
- Set environment variables:
  - (default) SAGEMAKER_ENDPOINT_NAME
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions).

Amazon SageMaker Endpoint

- Configure limited privilege to create SageMaker resources
- Deploy SageMaker model, endpointConfig, and endpoint
- Configure the SageMaker endpoint to be deployed in a VPC
- Deploy S3 VPC Endpoint and SageMaker Runtime VPC Interface

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:

@aws-solutions-constructs/aws-lambda-sagemakerendpoint

aws-lambda-secretsmanager

STABILITY EXPERIMENTAL

All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_lambda_secretsmanager`
Typescript	`@aws-solutions-constructs/aws-lambda-secretsmanager`
Java	`software.amazon.awsconstructs.services.lambdasecretsmanager`

Overview

This AWS Solutions Construct implements the AWS Lambda function and AWS Secrets Manager secret with the least privileged permissions.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { LambdaToSecretsmanagerProps, LambdaToSecretsmanager } from '@aws-solutions-constructs/aws-lambda-secretsmanager';
import * as lambda from 'aws-cdk-lib/aws-lambda';

const constructProps: LambdaToSecretsmanagerProps = {
  lambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_16_X,
    code: lambda.Code.fromAsset(`lambda`),
    handler: 'index.handler'
  },
};
```

```
new LambdaToSecretsmanager(this, 'test-lambda-secretsmanager-stack',
 constructProps);
```

Python

```
from aws_solutions_constructs.aws_lambda_secretsmanager import
 LambdaToSecretsmanagerProps, LambdaToSecretsmanager
from aws_cdk import (
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct


LambdaToSecretsmanager(
    self, 'test-lambda-secretsmanager-stack',
    lambda_function_props=_lambda.FunctionProps(
        code=_lambda.Code.from_asset('lambda'),
        runtime=_lambda.Runtime.PYTHON_3_9,
        handler='index.handler'
    )
)
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.lambdasecretsmanager.*;

new LambdaToSecretsmanager(this, "test-lambda-secretsmanager-stack", new
 LambdaToSecretsmanagerProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
secretProps?	`secretsmanager.SecretProps`	Optional user provided props to override the default props for Secrets Manager
existingSecretObj?	`secretsmanager.Secret`	Existing instance of Secrets Manager Secret object, If this is set then the secretProps is ignored
grantWriteAccess?	`string`	Optional Access granted to the Lambda function for the secret. "Read" or 'ReadWrite". Default is "Read"
secretEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the ARN of the secret. Default: SECRET_ARN.
existingVpc?	`ec2.IVpc`	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to

Name	Type	Description
		access network resources and an Interface Endpoint will be created in the VPC for AWS Secrets Manager. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the [`ec2.Vpc.fromLookup()`](https://docs.aws.amazon.com/cdk/api/v2/docs/aws-cdk-lib.aws_ec2.Vpc.html#static-fromwbrlookupscope-id-options) method.
vpcProps?	[`ec2.VpcProps`](https://docs.aws.amazon.com/cdk/api/v2/docs/aws-cdk-lib.aws_ec2.VpcProps.html)	Optional user-provided properties to override the default properties for the new VPC. `enableDnsHostnames`, `enableDnsSupport`, `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of lambda.Function created by the construct
secret	`secretsmanager.Secret`	Returns an instance of secretsmanager.Secret created by the construct
vpc?	`ec2.IVpc`	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS Lambda Function

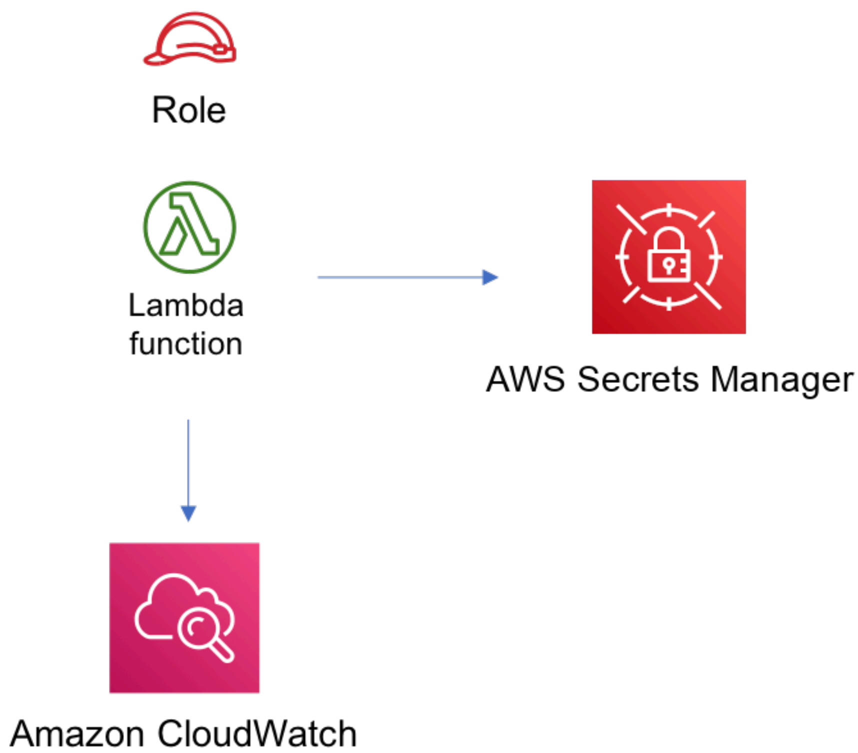
- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - (default) SECRET_ARN containing the ARN of the secret as return by CDK secretArn property.
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Amazon SecretsManager Secret

- Enable read-only access for the associated AWS Lambda Function
- Creates a new Secret

  - (default) random name
  - (default) random value
- Retain the Secret when deleting the CloudFormation stack

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-lambda-secretsmanager

aws-lambda-sns

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_lambda_sns`
Typescript	`@aws-solutions-constructs/aws-lambda-sns`
Java	`software.amazon.awsconstructs.services.lambdasns`

Overview

This AWS Solutions Construct implements an AWS Lambda function connected to an Amazon SNS topic.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { LambdaToSns, LambdaToSnsProps } from "@aws-solutions-constructs/aws-lambda-sns";
import * as lambda from 'aws-cdk-lib/aws-lambda';

new LambdaToSns(this, 'test-lambda-sns', {
  lambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler',
    code: lambda.Code.fromAsset(`lambda`)
  }
});
```

Python

```
from aws_solutions_constructs.aws_lambda_sns import LambdaToSns
from aws_cdk import (
```

```
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

LambdaToSns(
    self, 'test-lambda-sns-stack',
    lambda_function_props=_lambda.FunctionProps(
        code=_lambda.Code.from_asset('lambda'),
        runtime=_lambda.Runtime.PYTHON_3_9,
        handler='index.handler'
    )
)
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.lambdasns.*;

new LambdaToSns(this, "test-lambda-sns-stack", new LambdaToSnsProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFun`

Name	Type	Description
		`ctionProps` will cause an error.
lambdaFunctionProps?	[`lambda.FunctionProps`]()	User provided props to override the default props for the Lambda function.
existingTopicObj?	[`sns.Topic`]()	Existing instance of SNS Topic object, providing both this and `topicProps` will cause an error.
topicProps?	[`sns.TopicProps`]()	Optional user provided properties to override the default properties for the SNS topic.
existingVpc?	[`ec2.IVpc`]()	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to access network resources and an Interface Endpoint will be created in the VPC for Amazon SNS. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the [`ec2.Vpc.fromLookup()`]() method.

Name	Type	Description
vpcProps?	[`ec2.VpcProps`](#)	Optional user-provided properties to override the default properties for the new VPC. `enableDnsHostnames` , `enableDnsSupport` , `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:
topicArnEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the arn of the topic. Default: SNS_TOPIC_ARN
topicNameEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the name of the topic. Default: SNS_TOPIC_NAME

Name	Type	Description
enableEncryptionWi thCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the SNS Topic is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined: topicProp s.masterKey, encryptionKey or encryptionKeyProps.
encryptionKey?	[`kms.Key`]	An optional, imported encryptio n key to encrypt the SNS Topic with.
encryptionKeyProps?	[`kms.KeyProps`]	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SNS Topic with.

Pattern Properties

Name	Type	Description
lambdaFunction	[`lambda.Function`]	Returns an instance of the Lambda function created by the pattern.
snsTopic	[`sns.Topic`]	Returns an instance of the SNS topic created by the pattern.
vpc?	[`ec2.IVpc`]	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by

Name	Type	Description
		the pattern or the VPC supplied to the pattern constructor.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function to access the Firehose Delivery Stream
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - (default) SNS_TOPIC_NAME
  - (default) SNS_TOPIC_ARN
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Amazon SNS Topic

- Configure least privilege access permissions for SNS Topic
- Enable server-side encryption forSNS Topic using AWS managed KMS Key
- Enforce encryption of data in transit

Architecture

GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-lambda-sns

aws-lambda-sqs-lambda

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_lambda_sqs_lambda`
Typescript	`@aws-solutions-constructs/aws-lambda-sqs-lambda`

Language	Package
Java	software.amazon.awsconstructs.services.lambdasqslambda

Overview

This AWS Solutions Construct implements (1) an AWS Lambda function that is configured to send messages to a queue; (2) an Amazon SQS queue; and (3) an AWS Lambda function configured to consume messages from the queue.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { LambdaToSqsToLambda, LambdaToSqsToLambdaProps } from "@aws-solutions-constructs/aws-lambda-sqs-lambda";
import * as lambda from 'aws-cdk-lib/aws-lambda';

new LambdaToSqsToLambda(this, 'LambdaToSqsToLambdaPattern', {
  producerLambdaFunctionProps: {
      runtime: lambda.Runtime.NODEJS_16_X,
      handler: 'index.handler',
      code: lambda.Code.fromAsset(`producer-lambda`)
  },
  consumerLambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler',
    code: lambda.Code.fromAsset(`consumer-lambda`)
  }
});
```

Python

```
from aws_solutions_constructs.aws_lambda_sqs_lambda import LambdaToSqsToLambda
from aws_cdk import (
```

```
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

LambdaToSqsToLambda(
    self, 'LambdaToSqsToLambdaPattern',
    producer_lambda_function_props=_lambda.FunctionProps(
        code=_lambda.Code.from_asset('producer_lambda'),
        runtime=_lambda.Runtime.PYTHON_3_9,
        handler='index.handler'
    ),
    consumer_lambda_function_props=_lambda.FunctionProps(
        code=_lambda.Code.from_asset('consumer_lambda'),
        runtime=_lambda.Runtime.PYTHON_3_9,
        handler='index.handler'
    )
)
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.lambdasqslambda.*;

new LambdaToSqsToLambda(this, "LambdaToSqsToLambdaPattern", new
 LambdaToSqsToLambdaProps.Builder()
        .producerLambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("producer-lambda"))
                .handler("index.handler")
                .build())
        .consumerLambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("consumer-lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingProducerLambdaObj?	[lambda.Function](#)	An optional, existing Lambda function to be used instead of the default function for sending messages to the queue. Providing both this and `producerLambdaFunctionProps` will cause an error.
producerLambdaFunctionProps?	[lambda.FunctionProps](#)	Optional user-provided properties to override the default properties for the producer Lambda function.
existingQueueObj?	[sqs.Queue](#)	An optional, existing SQS queue to be used instead of the default queue. Providing both this and `queueProps` will cause an error.
queueProps?	[sqs.QueueProps](#)	Optional user-provided properties to override the default properties for the SQS queue. Providing both this and `existingQueueObj` will cause an error.
deployDeadLetterQueue?	`boolean`	Whether to create a secondary queue to be used as a dead letter queue. Defaults to `true`.
deadLetterQueueProps?	[sqs.QueueProps](#)	Optional user-provided props to override the default props for the dead letter queue.

Name	Type	Description
		Only used if the `deployDeadLetterQueue` property is set to `true`.
maxReceiveCount?	`number`	The number of times a message can be unsuccessfully dequeued before being moved to the dead letter queue. Defaults to 15.
existingConsumerLambdaObj?	[`lambda.Function`]()	An optional, existing Lambda function to be used instead of the default function for receiving/consuming messages from the queue. Providing both this and `consumerLambdaFunctionProps` will cause an error.
consumerLambdaFunctionProps?	[`lambda.FunctionProps`]()	Optional user-provided properties to override the default properties for the consumer Lambda function.
queueEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the URL of the queue. Default: SQS_QUEUE_URL
sqsEventSourceProps?	[`SqsEventSourceProps`]()	Optional user provided properties for the queue event source.

Name	Type	Description
existingVpc?	`ec2.IVpc`	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to access network resources and an Interface Endpoint will be created in the VPC for Amazon SQS. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the `ec2.Vpc.fromLookup()` method.
vpcProps?	`ec2.VpcProps`	Optional user-provided properties to override the default properties for the new VPC. `enableDnsHostnames` , `enableDnsSupport` , `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.

Name	Type	Description
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:

Pattern Properties

Name	Type	Description
producerLambdaFunction	`lambda.Function`	Returns an instance of the producer Lambda function created by the pattern.
sqsQueue	`sqs.Queue`	Returns an instance of the SQS queue created by the pattern.
deadLetterQueue?	`sqs.Queue | undefined`	Returns an instance of the dead letter queue created by the pattern, if one is deployed.
consumerLambdaFunction	`lambda.Function`	Returns an instance of the consumer Lambda function created by the pattern.
vpc?	`ec2.IVpc`	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

Default Settings

Out-of-the-box implementation of this Construct (without any overridden properties) will adhere to the following defaults:

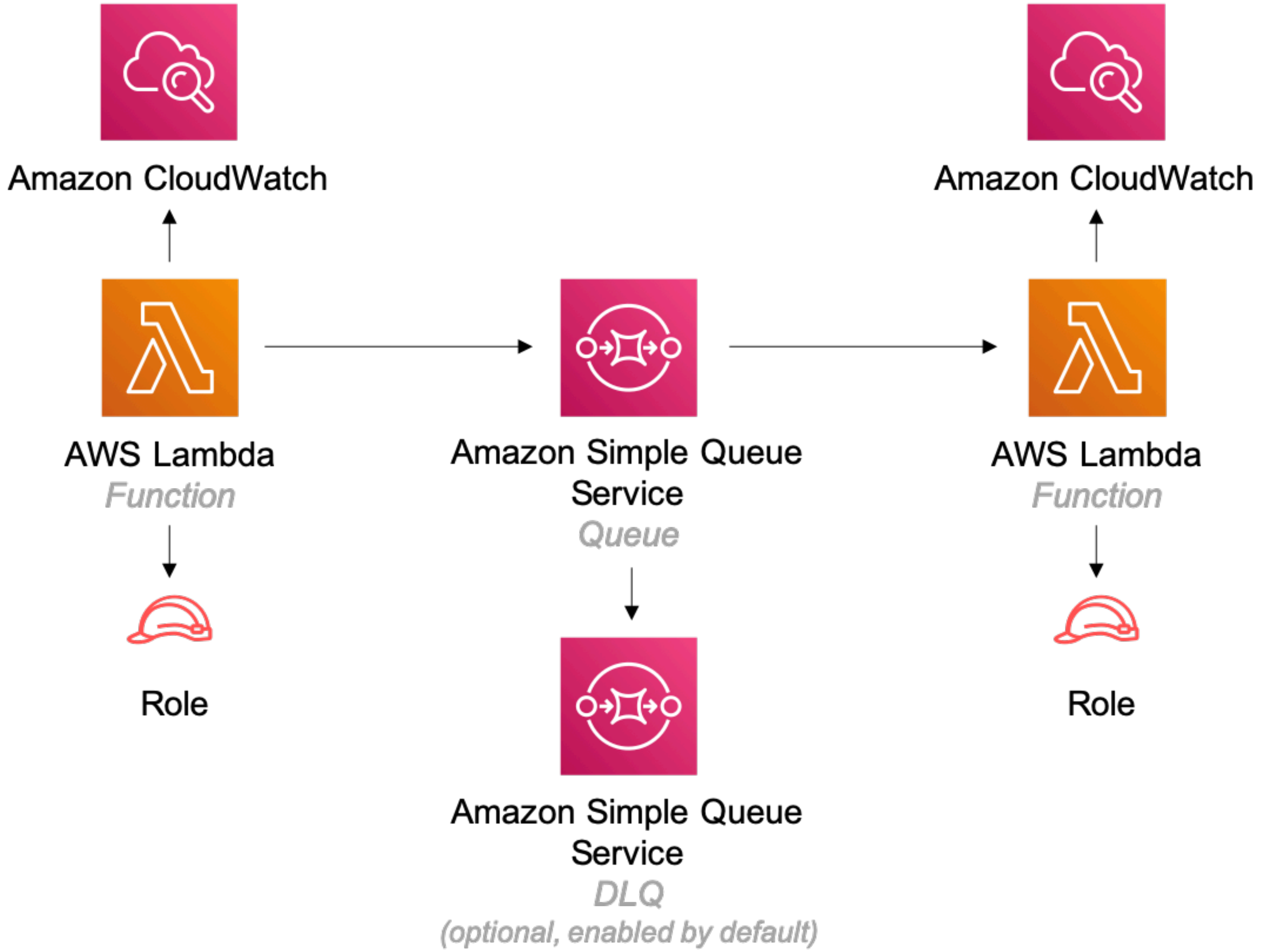
AWS Lambda Functions

- Configure limited privilege access IAM role for Lambda functions.
- Enable reusing connections with Keep-Alive for NodeJs Lambda functions.
- Enable X-Ray Tracing
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Amazon SQS Queue

- Deploy a dead letter queue for the primary queue.
- Enable server-side encryption for the primary queue using an AWS Managed KMS Key.
- Enforce encryption of data in transit

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-lambda-sqs-lambda]()

aws-lambda-sqs

CFN-RESOURCES **STABLE**

Language	Package
Python	`aws_solutions_constructs.aws_lambda_sqs`
Typescript	`@aws-solutions-constructs/aws-lambda-sqs`
Java	`software.amazon.awsconstructs.services.lambdasqs`

Overview

This AWS Solutions Construct implements an AWS Lambda function connected to an Amazon SQS queue.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { LambdaToSqs, LambdaToSqsProps } from "@aws-solutions-constructs/aws-lambda-sqs";
import * as lambda from 'aws-cdk-lib/aws-lambda';

new LambdaToSqs(this, 'LambdaToSqsPattern', {
  lambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler',
    code: lambda.Code.fromAsset(`lambda`)
  }
});
```

Python

```
from aws_solutions_constructs.aws_lambda_sqs import LambdaToSqs
from aws_cdk import (
```

```
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

LambdaToSqs(
    self, 'test-lambda-sqs-stack',
    lambda_function_props=_lambda.FunctionProps(
        code=_lambda.Code.from_asset('lambda'),
        runtime=_lambda.Runtime.PYTHON_3_9,
        handler='index.handler'
    )
)
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.lambdasqs.*;

new LambdaToSqs(this, "test-lambda-sqs-stack", new LambdaToSqsProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	An optional, existing Lambda function to be used instead of the default function. Providing both this and `lambdaFun`

Name	Type	Description
		`ctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	Optional user-provided properties to override the default properties for the Lambda function.
existingQueueObj?	`sqs.Queue`	An optional, existing SQS queue to be used instead of the default queue. Providing both this and `queueProps` will cause an error.
queueProps?	`sqs.QueueProps`	Optional user-provided properties to override the default properties for the SQS queue.
enableQueuePurging?	`boolean`	Whether to grant additiona l permissions to the Lambda function enabling it to purge the SQS queue. Defaults to `false`.
deployDeadLetterQueue?	`boolean`	Whether to create a secondary queue to be used as a dead letter queue. Defaults to `true`.
deadLetterQueueProps?	`sqs.QueueProps`	Optional user-provided props to override the default props for the dead letter queue. Only used if the `deployDeadLetterQueue` property is set to true.

Name	Type	Description
maxReceiveCount?	`number`	The number of times a message can be unsuccessfully dequeued before being moved to the dead letter queue. Defaults to 15.
existingVpc?	[`ec2.IVpc`]()	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to access network resources and an Interface Endpoint will be created in the VPC for Amazon SQS. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the [`ec2.Vpc.fromLookup()`]() method.
vpcProps?	[`ec2.VpcProps`]()	Optional user-provided properties to override the default properties for the new VPC. `enableDnsHostnames`, `enableDnsSupport`, `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.

Name	Type	Description
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:
queueEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the URL of the queue. Default: SQS_QUEUE_URL
enableEncryptionWithCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the queue is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined: queueProps.encryptionMasterKey, encryptionKey or encryptionKeyProps.
encryptionKey?	[`kms.Key`]()	An optional, imported encryption key to encrypt the SQS Queue with.
encryptionKeyProps?	[`kms.KeyProps`]()	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SQS queue with.

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of the Lambda function created by the pattern.
sqsQueue	`sqs.Queue`	Returns an instance of the SQS queue created by the pattern.
deadLetterQueue?	`sqs.Queue`	Returns an instance of the dead letter queue created by the pattern, if one is deployed.
vpc?	`ec2.IVpc`	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

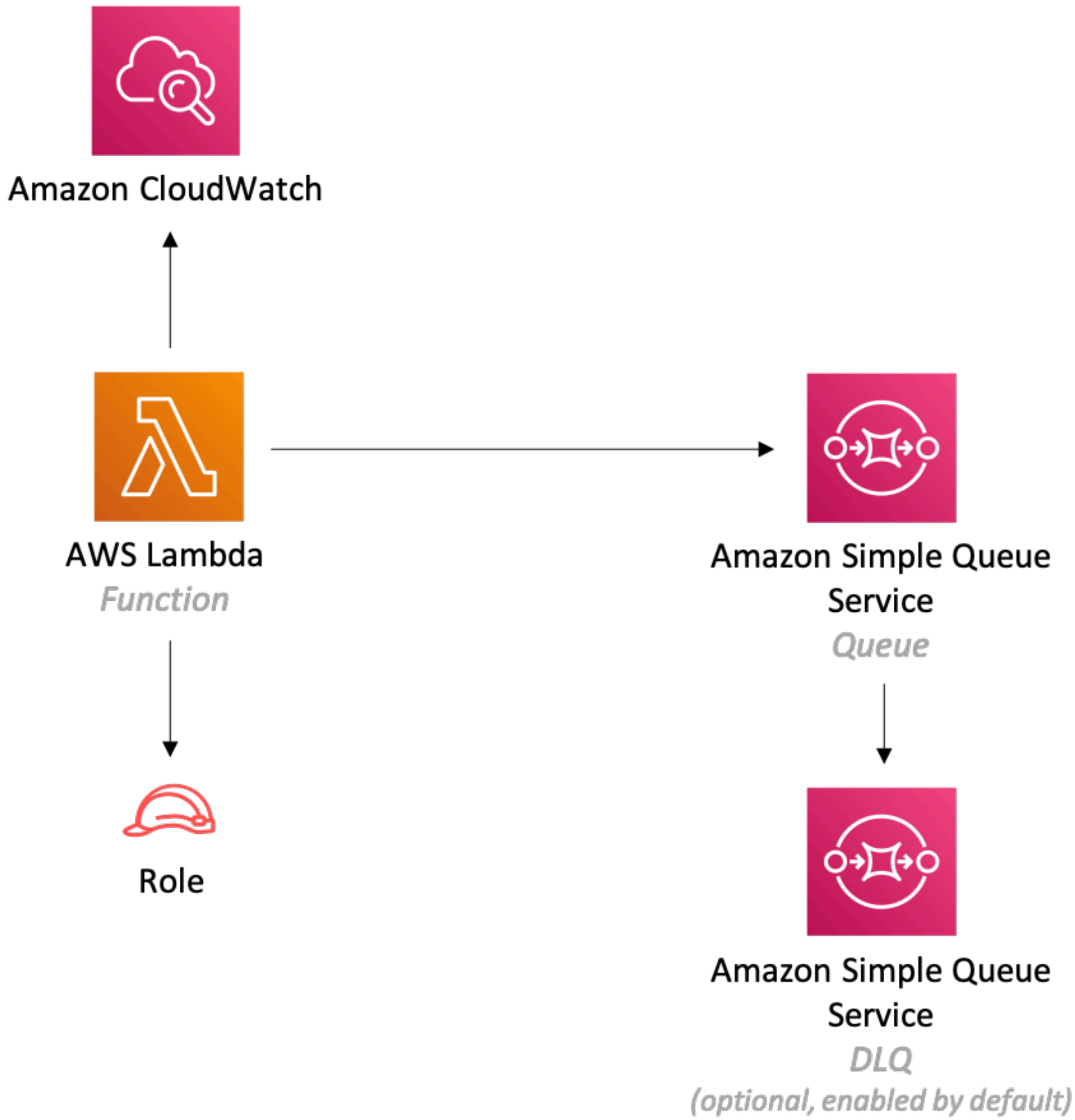
AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function.
- Enable reusing connections with Keep-Alive for NodeJs Lambda function.
- Allow the function to send messages only to the queue (purging can be enabled using the `enableQueuePurge` property).
- Enable X-Ray Tracing
- Set Environment Variables
  - SQS_QUEUE_URL
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Amazon SQS Queue

- Deploy SQS dead-letter queue for the source SQS Queue.
- Enable server-side encryption for source SQS Queue using AWS Managed KMS Key.
- Enforce encryption of data in transit

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-lambda-sqs

aws-lambda-ssmstringparameter

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_lambda_ssm_string_parameter`
Typescript	`@aws-solutions-constructs/aws-lambda-ssmstringparameter`
Java	`software.amazon.awsconstructs.services.lambdassmstringparameter`

Overview

This AWS Solutions Construct implements the AWS Lambda function and AWS Systems Manager Parameter Store String parameter with the least privileged permissions.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { LambdaToSsmstringparameterProps,  LambdaToSsmstringparameter } from '@aws-
solutions-constructs/aws-lambda-ssmstringparameter';
import * as lambda from 'aws-cdk-lib/aws-lambda';

const constructProps: LambdaToSsmstringparameterProps = {
  lambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_16_X,
    code: lambda.Code.fromAsset(`lambda`),
    handler: 'index.handler'
  },
  stringParameterProps: { stringValue: "test-string-value" }
};

new LambdaToSsmstringparameter(this, 'test-lambda-ssmstringparameter-stack',
 constructProps);
```

Python

```
from aws_solutions_constructs.aws_lambda_ssmstringparameter import
 LambdaToSsmstringparameter
from aws_cdk import (
    aws_lambda as _lambda,
    aws_ssm as ssm,
    Stack
)
from constructs import Construct

LambdaToSsmstringparameter(
    self, 'test-lambda-ssmstringparameter-stack',
    lambda_function_props=_lambda.FunctionProps(
        code=_lambda.Code.from_asset('lambda'),
        runtime=_lambda.Runtime.PYTHON_3_9,
        handler='index.handler'
    ),
    string_parameter_props=ssm.StringParameterProps(
        string_value="test-string-value")
)
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awscdk.services.ssm.*;
import software.amazon.awsconstructs.services.lambdassmstringparameter.*;

new LambdaToSsmstringparameter(this, "test-lambda-ssmstringparameter-stack",
        new LambdaToSsmstringparameterProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_16_X)
                        .code(Code.fromAsset("lambda"))
                        .handler("index.handler")
                        .build())
                .stringParameterProps(new StringParameterProps.Builder()
                        .stringValue("test-string-value")
                        .build())
                .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	[lambda.Function](#)	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	[lambda.FunctionProps](#)	User provided props to override the default props for the Lambda function.
existingStringParameterObj?	[ssm.StringParameter](#)	Existing instance of SSM String parameter object, providing

Name	Type	Description
		both this and `stringParameterProps` will cause an error
stringParameterProps?	[`ssm.StringParameterProps`]()	Optional user provided props to override the default props for SSM String parameter. If existingStringParameterObj is not set stringParameterProps is required. The only supported [`ssm.StringParameterProps.type`]() is [`STRING`]() if a different value is provided it will be overridden.
stringParameterEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the name of the parameter. Default: SSM_STRING_PARAMETER_NAME

Name	Type	Description
existingVpc?	`ec2.IVpc`	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function will use ENIs in the VPC to access network resources and an Interface Endpoint will be created in the VPC for AWS Systems Manager Parameter. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the `ec2.Vpc.fromLookup()` method.
vpcProps?	`ec2.VpcProps`	Optional user-provided properties to override the default properties for the new VPC. `enableDnsHostnames`, `enableDnsSupport`, `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.

Name	Type	Description
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:
stringParameterPer missions	`string`	Optional SSM String parameter permissions to grant to the Lambda function. One of the following may be specified: "Read", "ReadWrite".

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of lambda.Function created by the construct
stringParameter	`ssm.StringParameter`	Returns an instance of ssm.Strin gParameter created by the construct
vpc?	`ec2.IVpc`	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

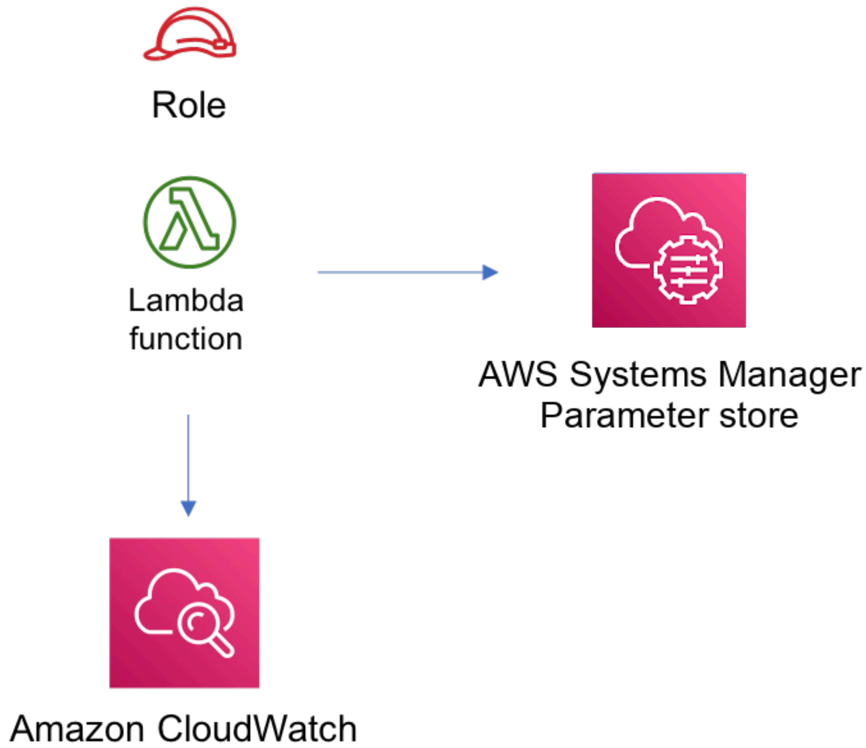
AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - (default) SSM_STRING_PARAMETER_NAME
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Amazon AWS Systems Manager Parameter Store String

- Enable read-only access for the associated AWS Lambda Function
- Creates a new SSM String parameter with the values provided
- Retain the SSM String parameter when deleting the CloudFormation stack

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-lambda-ssmstringparameter

aws-lambda-stepfunctions

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_lambda_stepfunctions`

Language	Package
Typescript	`@aws-solutions-constructs/aws-lambda-stepfunctions`
Java	`software.amazon.awsconstructs.services.lambdastepfunctions`

Overview

This AWS Solutions Construct implements an AWS Lambda function connected to an AWS Step Functions.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { LambdaToStepfunctions } from '@aws-solutions-constructs/aws-lambda-stepfunctions';
import * as stepfunctions from 'aws-cdk-lib/aws-stepfunctions';
import * as lambda from 'aws-cdk-lib/aws-lambda';

const startState = new stepfunctions.Pass(this, 'StartState');

new LambdaToStepfunctions(this, 'LambdaToStepfunctionsPattern', {
  lambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler',
    code: lambda.Code.fromAsset(`lambda`)
  },
  stateMachineProps: {
    definition: startState
  }
});
```

Python

```
from aws_solutions_constructs.aws_lambda_stepfunctions import LambdaToStepfunctions
from aws_cdk import (
    aws_lambda as _lambda,
    aws_stepfunctions as stepfunctions,
    Stack
)
from constructs import Construct

start_state = stepfunctions.Pass(self, 'start_state')

LambdaToStepfunctions(
    self, 'test-lambda-stepfunctions-stack',
    lambda_function_props=_lambda.FunctionProps(
        code=_lambda.Code.from_asset('lambda'),
        runtime=_lambda.Runtime.PYTHON_3_9,
        handler='index.handler'
    ),
    state_machine_props=stepfunctions.StateMachineProps(
        definition=start_state)
)
```

Java

```

import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awscdk.services.stepfunctions.*;
import software.amazon.awsconstructs.services.lambdastepfunctions.*;

final Pass startState = new Pass(this, "StartState");

new LambdaToStepfunctions(this, "test-lambda-stepfunctions-stack",
        new LambdaToStepfunctionsProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_16_X)
                        .code(Code.fromAsset("lambda"))
                        .handler("index.handler")
                        .build())
                .stateMachineProps(new StateMachineProps.Builder()
```

```
                    .definition(startState)
                    .build())
            .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
stateMachineProps	`sfn.StateMachineProps`	User provided props for the sfn.StateMachine.
createCloudWatchAlarms	`boolean`	Whether to create recommended CloudWatch alarms
logGroupProps?	`logs.LogGroupProps`	User provided props to override the default props for for the CloudWatchLogs LogGroup.
stateMachineEnvironmentVariableName?	`string`	Optional Name for the Lambda function environment variable set to the ARN of the state machine. Default: STATE_MACHINE_ARN
existingVpc?	`ec2.IVpc`	An optional, existing VPC into which this pattern should be deployed. When deployed in a VPC, the Lambda function

Name	Type	Description
		will use ENIs in the VPC to access network resources and an Interface Endpoint will be created in the VPC for Amazon Step Functions. If an existing VPC is provided, the `deployVpc` property cannot be `true`. This uses `ec2.IVpc` to allow clients to supply VPCs that exist outside the stack using the [`ec2.Vpc.fromLookup()`]() method.
vpcProps?	[`ec2.VpcProps`]()	Optional user-provided properties to override the default properties for the new VPC. `enableDnsHostnames`, `enableDnsSupport`, `natGateways` and `subnetConfiguration` are set by the pattern, so any values for those properties supplied here will be overridden. If `deployVpc` is not `true` then this property will be ignored.
deployVpc?	`boolean`	Whether to create a new VPC based on `vpcProps` into which to deploy this pattern. Setting this to true will deploy the minimal, most private VPC to run the pattern:

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of the Lambda function created by the pattern.
stateMachine	`sfn.StateMachine`	Returns an instance of StateMachine created by the construct.
stateMachineLogGroup	`logs.ILogGroup`	Returns an instance of the ILogGroup created by the construct for StateMachine
cloudwatchAlarms?	`cloudwatch.Alarm[]`	Returns a list of alarms created by the construct.
vpc?	`ec2.IVpc`	Returns an interface on the VPC used by the pattern (if any). This may be a VPC created by the pattern or the VPC supplied to the pattern constructor.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

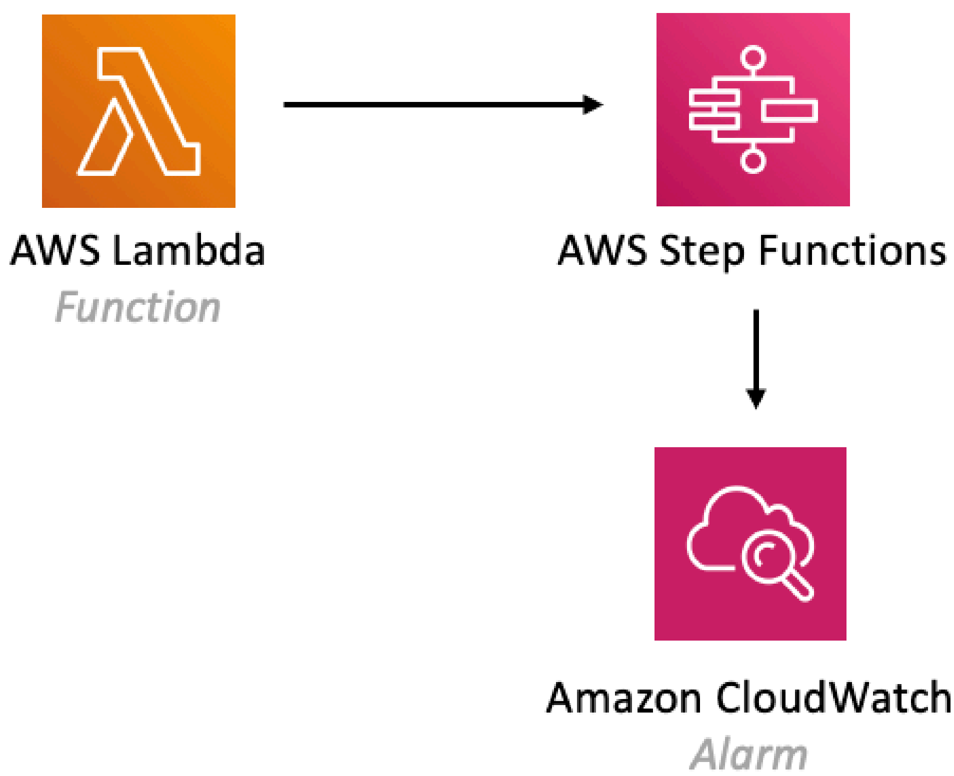
AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - (default) STATE_MACHINE_ARN
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

AWS Step Functions

- Enable CloudWatch logging for API Gateway
- Deploy best practices CloudWatch Alarms for the Step Functions

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-lambda-stepfunctions

aws-openapigateway-lambda

STABILITY EXPERIMENTAL

Language	Package
Python	aws_solutions_constructs.aws_openapigateway_lambda
Typescript	@aws-solutions-constructs/aws-openapigateway-lambda
Java	software.amazon.awsconstructs.services.openapigatewaylambda

Overview

This AWS Solutions Construct implements an Amazon API Gateway REST API defined by an OpenAPI specification file connected to an AWS Lambda function.

Here is a minimal deployable pattern definition.

NOTE The referenced openapi/apiDefinition.yaml openapi definition file and messages-lambda lambda package directory for the three code samples below can both be found under this constructs test folder (<repository_root>/source/patterns/@aws-solutions-constructs/aws-openapigateway-lambda/test)

Typescript

```
import { Stack, StackProps } from 'aws-cdk-lib';
import { Construct } from 'constructs';
import { OpenApiGatewayToLambda } from '@aws-solutions-constructs/aws-openapigateway-lambda';
import { Asset } from 'aws-cdk-lib/aws-s3-assets';
import * as path from 'path';
import * as lambda from 'aws-cdk-lib/aws-lambda';

const apiDefinitionAsset = new Asset(this, 'ApiDefinitionAsset', {
```

```
  path: path.join(__dirname, 'openapi/apiDefinition.yaml')
});

new OpenApiGatewayToLambda(this, 'OpenApiGatewayToLambda', {
  apiDefinitionAsset,
  apiIntegrations: [
    {
      id: 'MessagesHandler',
      lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_18_X,
        handler: 'index.handler',
        code: lambda.Code.fromAsset(`${__dirname}/messages-lambda`),
      }
    }
  ]
});
```

Python

```
from aws_cdk import (
    Stack,
    aws_s3_assets as s3_assets,
    aws_lambda as lambda_,
)
from constructs import Construct
from aws_solutions_constructs.aws_openapigateway_lambda import
 OpenApiGatewayToLambda, ApiIntegration

class TestStack(Stack):

    def __init__(self, scope: Construct, construct_id: str, **kwargs) -> None:
        super().__init__(scope, construct_id, **kwargs)

        api_definition_asset = s3_assets.Asset(self, "ApiDefinitionAsset", path="./
openapi/apiDefinition.yaml")

        api_integration = ApiIntegration(id="MessagesHandler",
 lambda_function_props={
            "runtime": lambda_.Runtime.NODEJS_18_X,
            "handler": "index.handler",
            "code": lambda_.Code.from_asset("./messages-lambda")
        })
```

```
        openapigateway_to_lambda = OpenApiGatewayToLambda(self,
            id="OpenApiGatewayToLambda",
            api_integrations=[api_integration],
            api_definition_asset=api_definition_asset
        )
```

Java

```
import software.amazon.awscdk.services.lambda.Code;
import software.amazon.awscdk.services.lambda.FunctionProps;
import software.amazon.awscdk.services.s3.assets.Asset;
import software.amazon.awscdk.services.s3.assets.AssetProps;
import software.amazon.awsconstructs.services.openapigatewaylambda.ApiIntegration;
import
 software.amazon.awsconstructs.services.openapigatewaylambda.OpenApiGatewayToLambda;
import
 software.amazon.awsconstructs.services.openapigatewaylambda.OpenApiGatewayToLambdaProps;
import software.constructs.Construct;
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;

import java.util.Collections;

import static software.amazon.awscdk.services.lambda.Runtime.NODEJS_18_X;

final Asset apiDefinitionAsset = new Asset(this, "ApiDefinition",
 AssetProps.builder().path("openapi/apiDefinition.yaml").build());

final ApiIntegration apiIntegration = ApiIntegration.builder()
    .id("MessagesHandler")
    .lambdaFunctionProps(new FunctionProps.Builder()
        .runtime(NODEJS_18_X)
        .code(Code.fromAsset("messages-lambda"))
        .handler("index.handler")
        .build())
    .build();

new OpenApiGatewayToLambda(this, "OpenApiGatewayToLambda",
 OpenApiGatewayToLambdaProps.builder()
    .apiDefinitionAsset(apiDefinitionAsset)
    .apiIntegrations(Collections.singletonList(apiIntegration))
```

```
    .build());
```

Pattern Construct Props

Name	Type	Description
apiGatewayProps?	[apigateway.RestApiBaseProps](#)	Optional user-provided props to override the default props for the API.
apiDefinitionBucket?	[s3.IBucket](#)	S3 Bucket where the OpenAPI spec file is located. When specifying this property, `apiDefinitionKey` must also be specified.
apiDefinitionKey?	`string`	S3 Object name of the OpenAPI spec file. When specifying this property, `apiDefinitionBucket` must also be specified.
apiDefinitionAsset?	[aws_s3_assets.Asset](#)	Local file asset of the OpenAPI spec file.
apiIntegrations	`ApiIntegration[]`	One or more key-value pairs that contain an id for the api integration and either an existing lambda function or an instance of the LambdaProps. Please see the `Overview of how the OpenAPI file transformation works` section below for more usage details.

Name	Type	Description
logGroupProps?	`logs.LogGroupProps`	User provided props to override the default props for for the CloudWatchLogs LogGroup.

Pattern Properties

Name	Type	Description
apiLambdaFunctions	`ApiLambdaFunction[]`	Returns an array of ApiLambda Function objects, where each has an id of the `apiIntegration` and the corresponding `lambda.Function` that it maps to.
apiGateway	`api.SpecRestApi`	Returns an instance of the API Gateway REST API created by the pattern.
apiGatewayCloudWatchRole?	`iam.Role`	Returns an instance of the iam.Role created by the construct for API Gateway for CloudWatch access.
apiGatewayLogGroup	`logs.LogGroup`	Returns an instance of the LogGroup created by the construct for API Gateway access logging to CloudWatch.

Overview of how the OpenAPI file transformation works

This construct automatically transforms an incoming OpenAPI Definition (residing locally or in S3) by auto-populating the `uri` fields of the `x-amazon-apigateway-integration` integrations with the resolved value of the backing lambda functions. It does so by allowing the user to specify the `apiIntegrations` property and then correlates it with the api definition.

Looking at an example - a user creates an instantiation of `apiIntegrations` that specifies one integration named `MessagesHandler` that passes in a set of `lambda.FunctionProps` and a second integration named `PhotosHandler` that passes in an existing `lambda.Function`:

```
const apiIntegrations: ApiIntegration[] = [
  {
    id: 'MessagesHandler',
    lambdaFunctionProps: {
      runtime: lambda.Runtime.NODEJS_18_X,
      handler: 'index.handler',
      code: lambda.Code.fromAsset(`${__dirname}/messages-lambda`),
    }
  },
  {
    id: 'PhotosHandler',
    existingLambdaObj: new lambda.Function(this, 'PhotosLambda', {
      runtime: lambda.Runtime.NODEJS_18_X,
      handler: 'index.handler',
      code: lambda.Code.fromAsset(`${__dirname}/photos-lambda`),
    })
  }
]
```

And a corresponding api definition with `GET` and `POST` methods on a `/messages` resource and a `GET` method on a `/photos` resource.

```
openapi: "3.0.1"
info:
  title: "api"
  version: "2023-02-20T20:46:08Z"
paths:
  /messages:
    get:
      x-amazon-apigateway-integration:
        httpMethod: "POST"
        uri: MessagesHandler
        passthroughBehavior: "when_no_match"
```

```
        type: "aws_proxy"
    post:
      x-amazon-apigateway-integration:
        httpMethod: "POST"
        uri: MessagesHandler
        passthroughBehavior: "when_no_match"
        type: "aws_proxy"
  /photos:
    get:
      x-amazon-apigateway-integration:
        httpMethod: "POST"
        uri: PhotosHandler
        passthroughBehavior: "when_no_match"
        type: "aws_proxy"
```

When the construct is created or updated, it will overwrite the `MessagesHandler` string with the fully resolved lambda proxy uri of the `MessagesHandlerLambdaFunction`, e.g., `arn:${Aws.PARTITION}:apigateway:${Aws.REGION}:lambda:path/2015-03-31/functions/${messagesLambda.functionArn}/invocations`, and similarly for the `PhotosHandler` string and `PhotosHandlerLambdaFunction`, resulting in a valid OpenAPI spec file that is then passed to the `SpecRestApi` construct.

For more information on specifying an API with OpenAPI, please see the [OpenAPI Specification]

ApiIntegration Details

This construct defines a custom type, `ApiIntegration`, that is specified as a required prop. The type has a required property, `id`, and two optional properties `existingLambdaObj` and `lambdaFunctionProps`. The `id` property is used to map the corresponding lambda function being defined with the placeholder string in the OpenAPI template file, and is not a CDK construct ID. Exactly one of `existingLambdaObj` or `lambdaFunctionProps` must be specified or the construct will throw an error.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon API Gateway

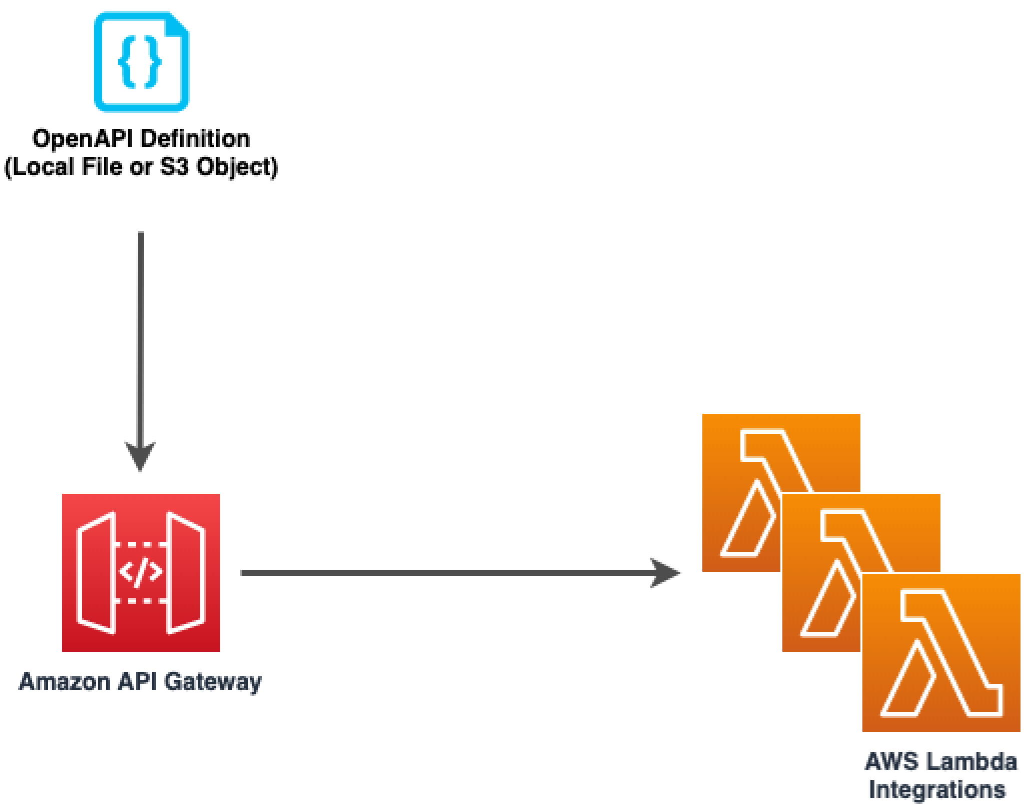
- Deploy an edge-optimized API endpoint

- Enable CloudWatch logging for API Gateway
- Configure least privilege access IAM role for API Gateway
- Enable X-Ray Tracing

AWS Lambda Function

- Configure limited privilege access IAM roles for Lambda functions
- Enable reusing connections with Keep-Alive for NodeJs Lambda functions
- Enable X-Ray Tracing
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-openapigateway-lambda

aws-route53-alb

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_route53_alb`
Typescript	`@aws-solutions-constructs/aws-route53-alb`
Java	`software.amazon.awsconstructs.services.route53alb`

Overview

This AWS Solutions Construct implements an Amazon Route53 Hosted Zone routing to an Application Load Balancer

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { Route53ToAlb } from '@aws-solutions-constructs/aws-route53-alb';

// Note - all alb constructs turn on ELB logging by default, so require that an
 environment including account
// and region be provided when creating the stack
//
// new MyStack(app, 'id', {env: {account: '123456789012', region: 'us-east-1' }});
new Route53ToAlb(this, 'Route53ToAlbPattern', {
  privateHostedZoneProps: {
    zoneName: 'www.example.com',
  },
  publicApi: false,
});
```

Python

```
from aws_solutions_constructs.aws_route53_alb import Route53ToAlb
from aws_cdk import (
    aws_route53 as route53,
    Stack
)
from constructs import Construct

# Note - all alb constructs turn on ELB logging by default, so require that an
 environment including account
# and region be provided when creating the stack
#
# MyStack(app, 'id', env=cdk.Environment(account='123456789012', region='us-
east-1'))
Route53ToAlb(self, 'Route53ToAlbPattern',
                public_api=False,
                private_hosted_zone_props=route53.HostedZoneProps(
                    zone_name='www.example.com',
                )
                )
```

Java

```
import software.constructs.Construct;
```

```
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.route53.*;
import software.amazon.awsconstructs.services.route53alb.*;

// Note - all alb constructs turn on ELB logging by default, so require that an
 environment including account
// and region be provided when creating the stack
//
// new MyStack(app, "id", StackProps.builder()
//         .env(Environment.builder()
//                 .account("123456789012")
//                 .region("us-east-1")
//                 .build());
new Route53ToAlb(this, "Route53ToAlbPattern",
        new Route53ToAlbProps.Builder()
                .privateHostedZoneProps(new HostedZoneProps.Builder()
                        .zoneName("www.example.com")
                        .build())
                .publicApi(false)
                .build());
```

Pattern Construct Props

This construct cannot create a new Public Hosted Zone, if you are creating a public API you must supply an existing Public Hosted Zone that will be reconfigured with a new Alias record. Public Hosted Zones are configured with public domain names and are not well suited to be launched and torn down dynamically, so this construct will only reconfigure existing Public Hosted Zones.

This construct can create Private Hosted Zones. If you want a Private Hosted Zone, then you can either provide an existing Private Hosted Zone or a privateHostedZoneProps value with at least the Domain Name defined.

Name	Type	Description
privateHostedZoneProps?	route53.PrivateHostedZonePr ops	Optional custom propertie s for a new Private Hosted Zone. Cannot be specified for a public API. Cannot specify

Name	Type	Description
		a VPC, it will use the VPC in existingVpc or the VPC created by the construct. Providing both this and existingHostedZone Interfaceis an error.
existingHostedZone Interface?	route53.IHostedZone	Existing Public or Private Hosted Zone (type must match publicApi setting). Specifyin g both this and privateHo stedZoneProps is an error. If this is a Private Hosted Zone, the associated VPC must be provided as the existingVpc property
loadBalancerProps?	elasticloadbalancingv2.Appl icationLoadBalancerProps	Optional custom propertie s for a new loadBalancer. Providing both this and existingLoadBalancer is an error. This cannot specify a VPC, it will use the VPC in existingV pc or the VPC created by the construct.
existingLoadBalancerObj?	elasticloadbalancingv2.Appl icationLoadBalancer	Existing Application Load Balancer to incorporate into the construct architect ure. Providing both this and loadBalancerProps is an error. The VPC containing this loadBalancer must match the VPC provided in existingVpc.

Name	Type	Description
vpcProps?	ec2.VpcProps	Optional custom properties for a VPC the construct will create. This VPC will be used by the new ALB and any Private Hosted Zone the construct creates (that's why loadBalan cerProps and privateHo stedZoneProps can't include a VPC). Providing both this and existingVpc is an error.
existingVpc?	ec2.IVpc	An existing VPC in which to deploy the construct. Providing both this and vpcProps is an error. If the client provides an existing load balancer and/or existing Private Hosted Zone, those constructs must exist in this VPC.
logAlbAccessLogs?	boolean	Whether to turn on Access Logs for the Application Load Balancer. Uses an S3 bucket with associated storage costs.Enabling Access Logging is a best practice. default - true
albLoggingBucketProps?	s3.BucketProps	Optional properties to customize the bucket used to store the ALB Access Logs. Supplying this and setting logAlbAccessLogs to false is an error. @default - none

publicApi | boolean | Whether the construct is deploying a private or public API. This has implications for the Hosted Zone, VPC and ALB. |

Pattern Properties

Name	Type	Description
hostedZone	route53.IHostedZone	The hosted zone used by the construct (whether created by the construct or providedb by the client)
vpc	ec2.IVpc	The VPC used by the construct (whether created by the construct or providedb by the client)
loadBalancer	elasticloadbalancingv2.ApplicationLoadBalancer	The Load Balancer used by the construct (whether created by the construct or providedb by the client)

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon Route53

- Adds an ALIAS record to the new or provided Hosted Zone that routes to the construct's ALB

Application Load Balancer

- Creates an Application Load Balancer with no Listener or target. The construct can incorporate an existing, fully configured ALB if provided.

Architecture

GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-route53-alb

aws-route53-apigateway

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic

Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_route53_apigateway
Typescript	@aws-solutions-constructs/aws-route53-apigateway
Java	software.amazon.awsconstructs.services.route53apigateway

Overview

This AWS Solutions Construct implements an Amazon Route 53 connected to a configured Amazon API Gateway REST API.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import * as route53 from "aws-cdk-lib/aws-route53";
import * as acm from "aws-cdk-lib/aws-certificatemanager";
import { Route53ToApiGateway } from '@aws-solutions-constructs/aws-route53-
apigateway';

// The construct requires an existing REST API, this can be created in raw CDK or
 extracted
// from a previously instantiated construct that created an API Gateway REST API
const existingRestApi = previouslyCreatedApigatewayToLambdaConstruct.apiGateway;

// domainName must match existing hosted zone in your account and the existing
 certificate
const ourHostedZone = route53.HostedZone.fromLookup(this, 'HostedZone', {
    domainName: "example.com",
```

```
});

const certificate = acm.Certificate.fromCertificateArn(
    this,
    "fake-cert",
    "arn:aws:acm:us-
east-1:123456789012:certificate/11112222-3333-1234-1234-123456789012"
);

// This construct can only be attached to a configured API Gateway.
new Route53ToApiGateway(this, 'Route53ToApiGatewayPattern', {
    existingApiGatewayInterface: existingRestApi,
    existingHostedZoneInterface: ourHostedZone,
    publicApi: true,
    existingCertificateInterface: certificate
});
```

Python

```
from aws_solutions_constructs.aws_route53_apigateway import Route53ToApiGateway
from aws_cdk import (
    aws_route53 as route53,
    aws_certificatemanager as acm,
    Stack
)
from constructs import Construct

# The construct requires an existing REST API, this can be created in raw CDK or
 extracted
# from a previously instantiated construct that created an API Gateway REST API
existingRestApi = previouslyCreatedApigatewayToLambdaConstruct.apiGateway

# domain_name must match existing hosted zone in your account and the existing
 certificate
ourHostedZone = route53.HostedZone.from_lookup(self, 'HostedZone',
                                               domain_name="example.com",
                                               )

# Obtain a pre-existing certificate from your account
certificate = acm.Certificate.from_certificate_arn(
    self,
    'existing-cert',
```

```
    "arn:aws:acm:us-
east-1:123456789012:certificate/11112222-3333-1234-1234-123456789012"
)

# This construct can only be attached to a configured API Gateway.
Route53ToApiGateway(self, 'Route53ToApigatewayPattern',
                    existing_api_gateway_interface=existingRestApi,
                    existing_hosted_zone_interface=ourHostedZone,
                    public_api=True,
                    existing_certificate_interface=certificate
                    )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.route53.*;
import software.amazon.awscdk.services.apigateway.*;
import software.amazon.awscdk.services.certificatemanager.*;
import software.amazon.awsconstructs.services.route53apigateway.*;

// The construct requires an existing REST API, this can be created in raw CDK
// or extracted from a previously instantiated construct that created an API
// Gateway REST API
final IRestApi existingRestApi =
 previouslyCreatedApigatewayToLambdaConstruct.getApiGateway();

// domainName must match existing hosted zone in your account and the existing
 certificate
final IHostedZone ourHostedZone = HostedZone.fromLookup(this, "HostedZone",
        new HostedZoneProviderProps.Builder()
                .domainName("example.com")
                .build());

// Obtain a pre-existing certificate from your account
final ICertificate certificate = Certificate.fromCertificateArn(
        this,
        "existing-cert",
        "arn:aws:acm:us-
east-1:123456789012:certificate/11112222-3333-1234-1234-123456789012");
```

```
// This construct can only be attached to a configured API Gateway.
new Route53ToApiGateway(this, "Route53ToApiGatewayPattern",
        new Route53ToApiGatewayProps.Builder()
                .existingApiGatewayInterface(existingRestApi)
                .existingHostedZoneInterface(ourHostedZone)
                .publicApi(true)
                .existingCertificateInterface(certificate)
                .build());
```

Pattern Construct Props

This construct cannot create a new Public Hosted Zone, if you are creating a public API you must supply an existing Public Hosted Zone that will be reconfigured with a new Alias record. Public Hosted Zones are configured with public domain names and are not well suited to be launched and torn down dynamically, so this construct will only reconfigure existing Public Hosted Zones.

This construct can create Private Hosted Zones. If you want a Private Hosted Zone, then you can either provide an existing Private Hosted Zone or a privateHostedZoneProps value with at least the Domain Name defined. If you are using privateHostedZoneProps, an existing wildcard certificate (*.example.com) must be issued from a previous domain to be used in the newly created Private Hosted Zone. New certificate creation and validation do not take place in this construct. A private Rest API already exists in a VPC, so that VPC must be provided in the existingVpc prop. There is no scenario where this construct can create a new VPC (since it can't create a new API), so the vpcProps property is not supported on this construct.

Name	Type	Description
publicApi	boolean	Whether the construct is deploying a private or public API. This has implications for the Hosted Zone and VPC.
privateHostedZoneProps?	route53.PrivateHostedZoneProps	Optional custom properties for a new Private Hosted Zone. Cannot be specified for a public API. Cannot specify a VPC, it will use the VPC in

Name	Type	Description
		existingVpc or the VPC created by the construct. Providing both this and existingHostedZone Interface is an error.
existingHostedZone Interface?	[route53.IHostedZone]()	Existing Public or Private Hosted Zone (type must match publicApi setting). Specifyin g both this and privateHo stedZoneProps is an error. If this is a Private Hosted Zone, the associated VPC must be provided as the existingVpc property.
existingVpc?	[ec2.IVpc]()	An existing VPC in which to deploy the construct.
existingApiGateway Interface	[api.IRestApi]()	The existing API Gateway instance that will be connected to the Route 53 hosted zone. *Note that Route 53 can only be connected to a configured API Gateway, so this construct only accepts an existing IRestApi and does not accept apiGatewa yProps.*
existingCertificateInterfac e	[certificatemanager.ICertificate]()	An existing AWS Certificate Manager certificate for your custom domain name.

Pattern Properties

Name	Type	Description
hostedZone	route53.IHostedZone	The hosted zone used by the construct (whether created by the construct or provided by the client)
vpc?	ec2.IVpc	The VPC used by the construct.
apiGateway	api.RestApi	Returns an instance of the API Gateway REST API created by the pattern.
certificate	certificatemanager.ICertificate	THe certificate used by the construct (whether create by the construct or provided by the client)

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

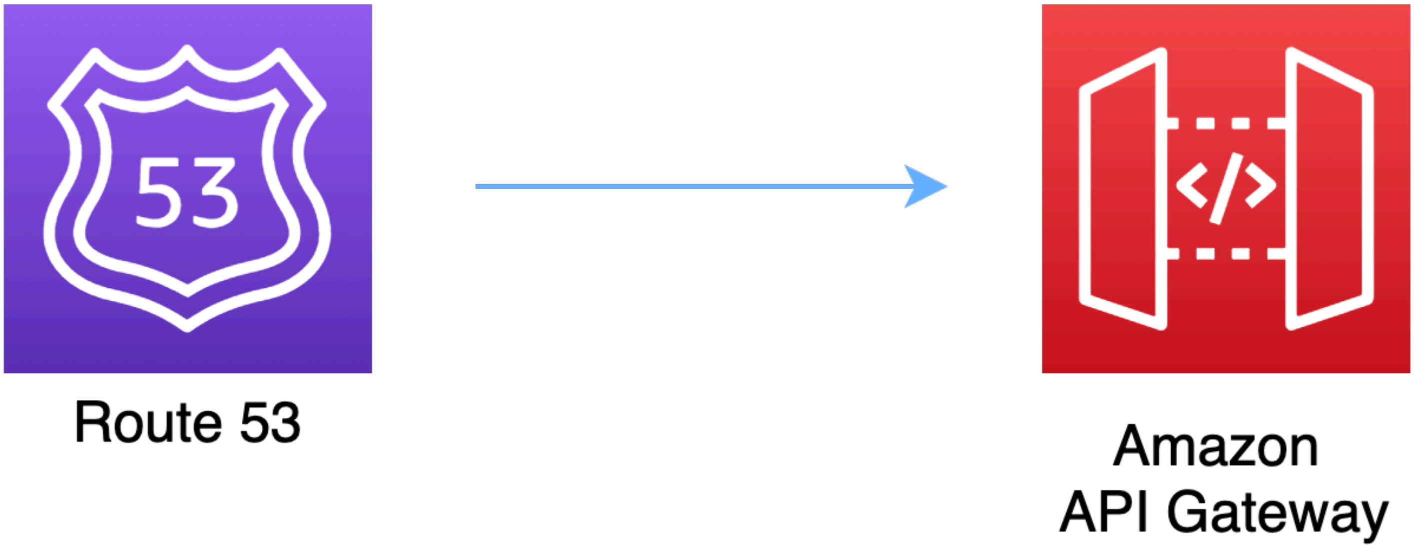
Amazon Route53

- Adds an ALIAS record to the new or provided Hosted Zone that routes to the construct's API Gateway

Amazon API Gateway

- User provided API Gateway object is used as-is
- Sets up custom domain name mapping to API

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	[@aws-solutions-constructs/aws-route53-apigateway]()

aws-s3-lambda

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_s3_lambda`
Typescript	`@aws-solutions-constructs/aws-s3-lambda`
Java	`software.amazon.awsconstructs.services.s3lambda`

Overview

This AWS Solutions Construct implements an Amazon S3 bucket connected to an AWS Lambda function.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import * as lambda from "aws-cdk-lib/aws-lambda";
import { S3ToLambdaProps, S3ToLambda } from '@aws-solutions-constructs/aws-s3-
lambda';

new S3ToLambda(this, 'test-s3-lambda', {
  lambdaFunctionProps: {
    code: lambda.Code.fromAsset(`lambda`),
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler'
  },
});
```

Python

```
from aws_solutions_constructs.aws_s3_lambda import S3ToLambda
from aws_cdk import (
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

S3ToLambda(self, 'test_s3_lambda',
           lambda_function_props=_lambda.FunctionProps(
               code=_lambda.Code.from_asset('lambda'),
               runtime=_lambda.Runtime.PYTHON_3_9,
               handler='index.handler'
           )
           )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.s3lambda.*;

new S3ToLambda(this, "test-s3-lambda'", new S3ToLambdaProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	Optional user provided props to override the default props for the Lambda function.
existingBucketObj?	`s3.Bucket`	Existing instance of S3 Bucket object. If this is provided, then also providing bucketProps is an error.

Name	Type	Description
bucketProps?	s3.BucketProps	Optional user provided props to override the default props for the S3 Bucket.
s3EventSourceProps?	S3EventSourceProps	Optional user provided props to override the default props for S3EventSourceProps
loggingBucketProps?	s3.BucketProps	Optional user provided props to override the default props for the S3 Logging Bucket.
logS3AccessLogs?	boolean	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with associated storage costs for the logs. Enabling Access Logging is a best practice. default - true

Pattern Properties

Name	Type	Description
lambdaFunction	lambda.Function	Returns an instance of the lambda.Function created by the construct
s3Bucket?	s3.Bucket	Returns an instance of the s3.Bucket created by the construct
s3LoggingBucket?	s3.Bucket	Returns an instance of s3.Bucket created by the construct as the logging bucket for the primary bucket.

Name	Type	Description
s3BucketInterface	`s3.IBucket`	Returns an instance of s3.IBucket created by the construct

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

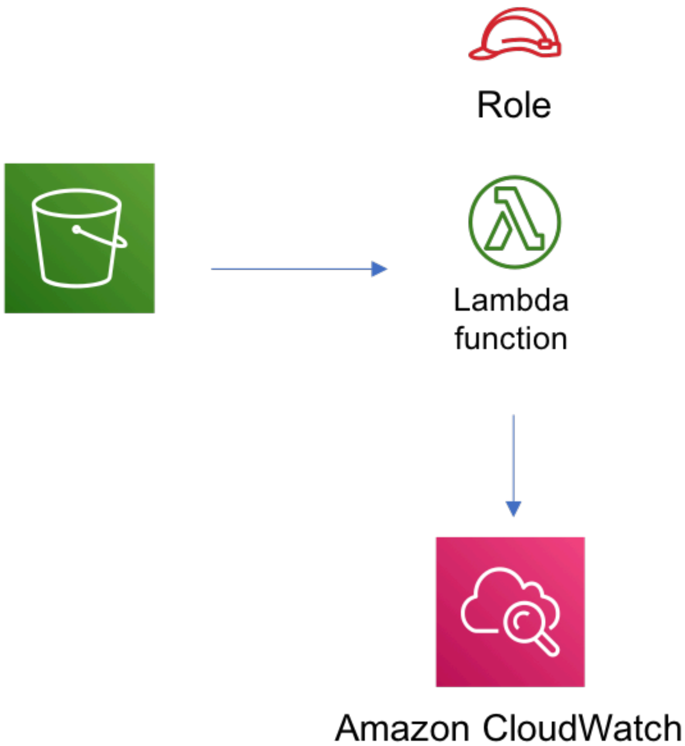
Amazon S3 Bucket

- Configure Access logging for S3 Bucket
- Enable server-side encryption for S3 Bucket using AWS managed KMS Key
- Enforce encryption of data in transit
- Turn on the versioning for S3 Bucket
- Don't allow public access for S3 Bucket
- Retain the S3 Bucket when deleting the CloudFormation stack
- Applies Lifecycle rule to move noncurrent object versions to Glacier storage after 90 days

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-s3-lambda

aws-s3-sns

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_s3_sns`
Typescript	`@aws-solutions-constructs/aws-s3-sns`
Java	`software.amazon.awsconstructs.services.s3sns`

Overview

This AWS Solutions Construct implements an Amazon S3 Bucket that is configured to send S3 event messages to an Amazon SNS topic.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { S3ToSns } from "@aws-solutions-constructs/aws-s3-sns";

new S3ToSns(this, 'S3ToSNSPattern', {});
```

Python

```
from aws_solutions_constructs.aws_s3_sns import S3ToSns
from aws_cdk import Stack
from constructs import Construct

S3ToSns(self, 'S3ToSNSPattern')
```

Java

```
import software.constructs.Construct;
```

```
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.s3sns.*;

new S3ToSns(this, "S3ToSNSPattern", new S3ToSnsProps.Builder()
        .build());
```

Pattern Construct Props

Name	Type	Description
existingBucketObj?	`s3.Bucket`	Existing instance of S3 Bucket object. If this is provided, then also providing bucketProps is an error.
bucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the S3 Bucket.
s3EventTypes?	`s3.EventType[]`	The S3 event types that will trigger the notification. Defaults to s3.EventType.OBJEC T_CREATED.
s3EventFilters?	`s3.NotificationKeyFilter[]`	S3 object key filter rules to determine which objects trigger this event. If not specified no filter rules will be applied.
loggingBucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the S3 Logging Bucket.
logS3AccessLogs?	`boolean`	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with

Name	Type	Description
		associated storage costs for the logs. Enabling Access Logging is a best practice. default - true
existingTopicObj?	[sns.Topic](#)	An optional, existing SNS topic to be used instead of the default topic. Providing both this and `topicProps` will cause an error. If the SNS Topic is encrypted with a Customer-Managed KMS Key, the key must be specified in the `existingTopicEncryptionKey` property.
existingTopicEncryptionKey?	[kms.Key](#)	If an existing topic is provided in the `existingTopicObj` property, and that topic is encrypted with a Customer-Managed KMS key, this property also needs to be set with same key.
topicProps?	[sns.TopicProps](#)	Optional user provided props to override the default props for the SNS topic.
enableEncryptionWithCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the topic is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined: topicProps.encryptionMasterKey, encryptionKey or encryptionKeyProps.

Name	Type	Description
encryptionKey?	kms.Key	An optional, imported encryption key to encrypt the SNS Topic with.
encryptionKeyProps?	kms.KeyProps	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SNS Topic with.

Pattern Properties

Name	Type	Description
snsTopic	sns.Topic	Returns an instance of the SNS Topic created by the pattern.
encryptionKey?	kms.Key	Returns an instance of the kms.Key associated with the SNS Topic
s3Bucket?	s3.Bucket	Returns an instance of the s3.Bucket created by the construct
s3LoggingBucket?	s3.Bucket	Returns an instance of s3.Bucket created by the construct as the logging bucket for the primary bucket.
s3BucketInterface	s3.IBucket	Returns an instance of s3.IBucket created by the construct.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon S3 Bucket

- Configure Access logging for the S3 Bucket
- Enable server-side encryption for S3 Bucket using an AWS managed KMS Key
- Enforce encryption of data in transit
- Turn on the versioning for the S3 Bucket
- Don't allow public access for the S3 Bucket
- Retain the S3 Bucket when deleting the CloudFormation stack
- Applies Lifecycle rule to move noncurrent object versions to Glacier storage after 90 days

Amazon SNS Topic

- Configure least privilege SNS Topic access policy to allow the S3 Bucket to publish messages to it
- Enable server-side encryption for the SNS Topic using an AWS managed KMS Key
- Enforce encryption of data in transit

Architecture

GitHub

To view the code for this pattern, create/view issues and pull requests, and more:

@aws-solutions-constructs/aws-s3-sns

aws-s3-sqs

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_s3_sqs`
Typescript	`@aws-solutions-constructs/aws-s3-sqs`
Java	`software.amazon.awsconstructs.services.s3sqs`

Overview

This AWS Solutions Construct implements an Amazon S3 Bucket that is configured to send notifications to an Amazon SQS queue.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { S3ToSqs } from "@aws-solutions-constructs/aws-s3-sqs";

new S3ToSqs(this, 'S3ToSQSPattern', {});
```

Python

```
from aws_solutions_constructs.aws_s3_sqs import S3ToSqs
from aws_cdk import Stack
from constructs import Construct

S3ToSqs(self, 'S3ToSQSPattern')
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.s3sqs.*;

new S3ToSqs(this, "S3ToSQSPattern", new S3ToSqsProps.Builder()
        .build());
```

Pattern Construct Props

Name	Type	Description
existingBucketObj?	`s3.Bucket`	Existing instance of S3 Bucket object. If this is provided, then also providing bucketProps is an error.
bucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the S3 Bucket.
s3EventTypes?	`s3.EventType[]`	The S3 event types that will trigger the notification. Defaults to s3.EventType.OBJECT_CREATED.
s3EventFilters?	`s3.NotificationKeyFilter[]`	S3 object key filter rules to determine which objects trigger

Name	Type	Description
		this event. If not specified no filter rules will be applied.
existingQueueObj?	[sqs.Queue](#)	Existing SQS queue to be used instead of the default queue. Providing both this and `queueProps` will cause an error. If the SQS queue is encrypted, the KMS key utilized for encryption must be a customer managed CMK.
queueProps?	[sqs.QueueProps](#)	Optional user provided props to override the default props for the SQS queue.
deadLetterQueueProps?	[sqs.QueueProps](#)	Optional user provided props to override the default props for the dead letter SQS queue.
deployDeadLetterQueue?	`boolean`	Whether to create a secondary queue to be used as a dead letter queue. Defaults to true.
maxReceiveCount?	`number`	The number of times a message can be unsuccessfully dequeued before being moved to the dead letter queue. Defaults to 15.

Name	Type	Description
enableEncryptionWithCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the queue is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined: queueProps.encryptionMasterKey, encryptionKey or encryptionKeyProps.
encryptionKey?	`kms.Key`	An optional, imported encryption key to encrypt the SQS Queue with.
encryptionKeyProps?	`kms.KeyProps`	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SQS queue with.
loggingBucketProps?	`s3.BucketProps`	Optional user provided props to override the default props for the S3 Logging Bucket.
logS3AccessLogs?	boolean	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with associated storage costs for the logs. Enabling Access Logging is a best practice. default - true

Pattern Properties

Name	Type	Description
sqsQueue	`sqs.Queue`	Returns an instance of the SQS queue created by the pattern.
deadLetterQueue?	`sqs.Queue`	Returns an instance of the dead-letter SQS queue created by the pattern.
encryptionKey	`kms.IKey`	Returns an instance of kms.Key used for the SQS queue.
s3Bucket?	`s3.Bucket`	Returns an instance of the s3.Bucket created by the construct
s3LoggingBucket?	`s3.Bucket`	Returns an instance of s3.Bucket created by the construct as the logging bucket for the primary bucket.
s3BucketInterface	`s3.IBucket`	Returns an instance of s3.IBucke t created by the construct.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon S3 Bucket

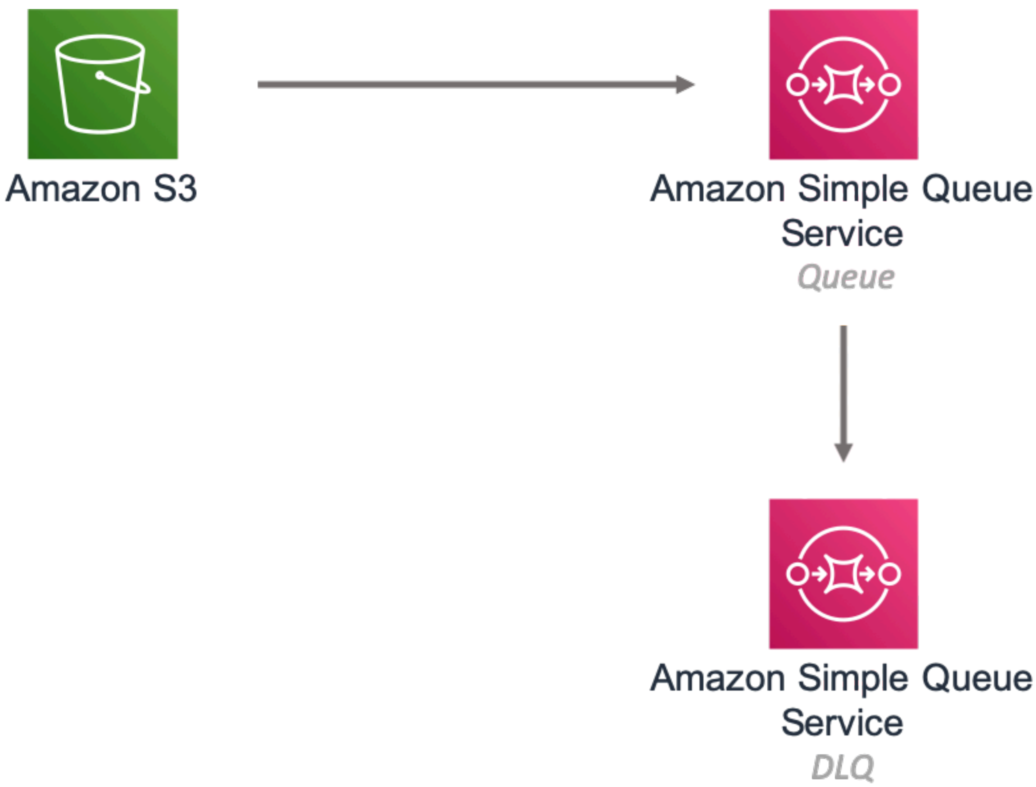
- Configure Access logging for S3 Bucket
- Enable server-side encryption for S3 Bucket using AWS managed KMS Key
- Enforce encryption of data in transit
- Turn on the versioning for S3 Bucket
- Don’t allow public access for S3 Bucket

- Retain the S3 Bucket when deleting the CloudFormation stack
- Applies Lifecycle rule to move noncurrent object versions to Glacier storage after 90 days

Amazon SQS Queue

- Configure least privilege access permissions for SQS Queue
- Deploy SQS dead-letter queue for the source SQS Queue
- Enable server-side encryption for SQS Queue using Customer managed KMS Key
- Enforce encryption of data in transit

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:

@aws-solutions-constructs/aws-s3-sqs

aws-s3-stepfunctions

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the [Semantic Versioning](https://semver.org/) model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_s3_stepfunctions`
Typescript	`@aws-solutions-constructs/aws-s3-stepfunctions`
Java	`software.amazon.awsconstructs.services.s3stepfunctions`

Overview

This AWS Solutions Construct implements an Amazon S3 bucket connected to an AWS Step Functions.

Note - This constructs sends S3 Event Notification to EventBridge, then triggers AWS Step Functions State Machine executions from EventBridge.

An alternative architecture can be built that triggers a Lambda function from S3 Event notifications using aws-s3-lambda and aws-lambda-stepfunctions. Channelling the S3 events through Lambda is less flexible than EventBridge, but is more cost effective and has lower latency.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
```

```
import { S3ToStepfunctions, S3ToStepfunctionsProps } from '@aws-solutions-
constructs/aws-s3-stepfunctions';
import * as stepfunctions from 'aws-cdk-lib/aws-stepfunctions';

const startState = new stepfunctions.Pass(this, 'StartState');

new S3ToStepfunctions(this, 'test-s3-stepfunctions-stack', {
    stateMachineProps: {
      definition: startState
    }
});
```

Python

```
from aws_solutions_constructs.aws_s3_stepfunctions import S3ToStepfunctions
from aws_cdk import (
    aws_stepfunctions as stepfunctions,
    Stack
)
from constructs import Construct

start_state = stepfunctions.Pass(self, 'start_state')

S3ToStepfunctions(
    self, 'test_s3_stepfunctions_stack',
    state_machine_props=stepfunctions.StateMachineProps(
        definition=start_state)
)
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.stepfunctions.*;
import software.amazon.awsconstructs.services.s3stepfunctions.*;

final Pass startState = new Pass(this, "StartState");

new S3ToStepfunctions(this, "test_s3_stepfunctions_stack",
```

```
        new S3ToStepfunctionsProps.Builder()
                .stateMachineProps(new StateMachineProps.Builder()
                        .definition(startState)
                        .build())
                .build());
```

Pattern Construct Props

Name	Type	Description
existingBucketObj?	[s3.IBucket](#)	Existing instance of S3 Bucket object. If this is provided, then also providing bucketProps is an error. **The existing bucket must have [EventBridge enabled](#) for this to work.**
bucketProps?	[s3.BucketProps](#)	Optional user provided props to override the default props for the S3 Bucket.
stateMachineProps	[sfn.StateMachineProps](#)	User provided props to override the default props for sfn.State Machine.
eventRuleProps?	[events.RuleProps](#)	Optional user provided eventRuleProps to override the defaults.
deployCloudTrail?	`boolean`	Whether to deploy a Trail in AWS CloudTrail to log API events in Amazon S3. Defaults to `true`. **This is now deprecate d and ignored because the construct no longer needs CloudTrail since it uses S3 Event Notifications**.

Name	Type	Description
createCloudWatchAlarms	`boolean`	Whether to create recommend ed CloudWatch alarms.
logGroupProps?	[logs.LogGroupProps](#)	Optional user provided props to override the default props for for the CloudWatchLogs LogGroup.
loggingBucketProps?	[s3.BucketProps](#)	Optional user provided props to override the default props for the S3 Logging Bucket.
logS3AccessLogs?	boolean	Whether to turn on Access Logging for the S3 bucket. Creates an S3 bucket with associated storage costs for the logs. Enabling Access Logging is a best practice. default - true

Pattern Properties

Name	Type	Description
stateMachine	[sfn.StateMachine](#)	Returns an instance of sfn.State Machine created by the construct.
stateMachineLogGroup	[logs.ILogGroup](#)	Returns an instance of the ILogGroup created by the construct for StateMachine.
cloudwatchAlarms?	[cloudwatch.Alarm[]](#)	Returns a list of cloudwatc h.Alarm created by the construct.

Name	Type	Description
s3Bucket?	[s3.Bucket](#)	Returns an instance of the s3.Bucket created by the construct.
s3LoggingBucket?	[s3.Bucket](#)	Returns an instance of s3.Bucket created by the construct as the logging bucket for the primary bucket.
s3BucketInterface	[s3.IBucket](#)	Returns an instance of s3.IBucke t created by the construct.

Note - with the release of Enable EventBridge for Amazon S3, AWS CloudTrail is no longer required to implement this construct. Because of this, the following properties have been removed: - cloudtrail - cloudtrailBucket - cloudtrailLoggingBucket

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon S3 Bucket

- Enable EventBridge to send events from the S3 Bucket
- Configure Access logging for S3 Bucket
- Enable server-side encryption for S3 Bucket using AWS managed KMS Key
- Enforce encryption of data in transit
- Turn on the versioning for S3 Bucket
- Don't allow public access for S3 Bucket
- Retain the S3 Bucket when deleting the CloudFormation stack
- Applies Lifecycle Rule to move noncurrent object versions to Glacier storage after 90 days

AWS S3 Event Notification

- Enable S3 to send events to EventBridge when an object is created.

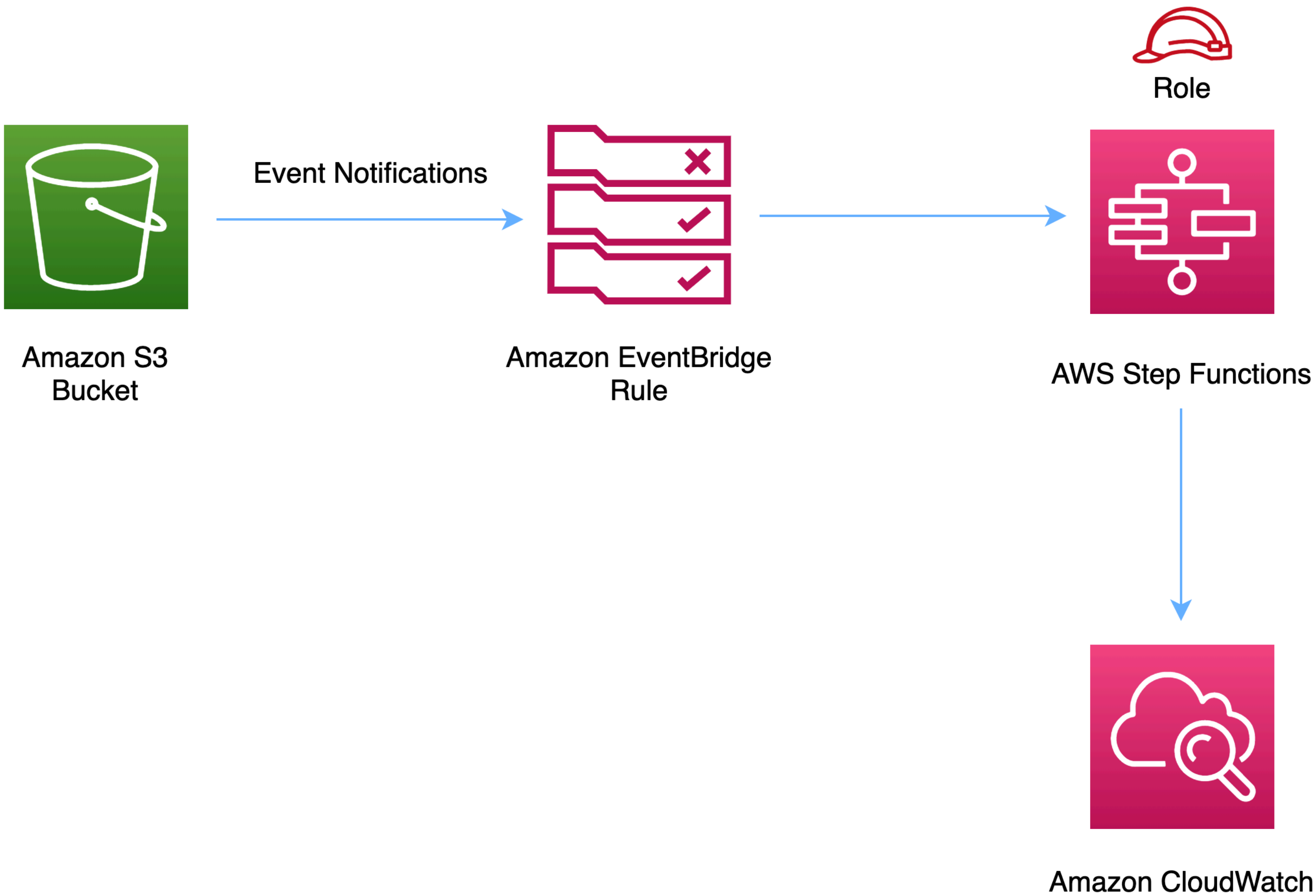
Amazon CloudWatch Events Rule

- Grant least privilege permissions to CloudWatch Events to trigger the Lambda Function

AWS Step Functions

- Enable CloudWatch logging for API Gateway
- Deploy best practices CloudWatch Alarms for the Step Functions

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-s3-stepfunctions

aws-sns-lambda

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_sns_lambda`
Typescript	`@aws-solutions-constructs/aws-sns-lambda`
Java	`software.amazon.awsconstructs.services.snslambda`

Overview

This AWS Solutions Construct implements an Amazon SNS connected to an AWS Lambda function.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { SnsToLambda, SnsToLambdaProps } from "@aws-solutions-constructs/aws-sns-lambda";
import * as lambda from 'aws-cdk-lib/aws-lambda';

new SnsToLambda(this, 'test-sns-lambda', {
```

```
  lambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler',
    code: lambda.Code.fromAsset(`lambda`)
  }
});
```

Python

```
from aws_solutions_constructs.aws_sns_lambda import SnsToLambda
from aws_cdk import (
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

SnsToLambda(self, 'test_sns_lambda',
            lambda_function_props=_lambda.FunctionProps(
                code=_lambda.Code.from_asset('lambda'),
                runtime=_lambda.Runtime.PYTHON_3_9,
                handler='index.handler'
            )
            )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.snslambda.*;

new SnsToLambda(this, "test-lambda-sqs-stack", new SnsToLambdaProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	`lambda.Function`	Existing instance of Lambda Function object, providing both this and `lambdaFunctionProps` will cause an error.
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
existingTopicObj?	`sns.Topic`	Existing instance of SNS Topic object, providing both this and `topicProps` will cause an error.
topicProps?	`sns.TopicProps`	Optional user provided properties to override the default properties for the SNS topic.
enableEncryptionWithCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the SNS Topic is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined: topicProps.masterKey, encryptionKey or encryptionKeyProps.
encryptionKey?	`kms.Key`	An optional, imported encryption key to encrypt the SNS Topic with.

Name	Type	Description
encryptionKeyProps?	`kms.KeyProps`	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SNS Topic with.

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of the Lambda function created by the pattern.
snsTopic	`sns.Topic`	Returns an instance of the SNS topic created by the pattern.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

Amazon SNS Topic

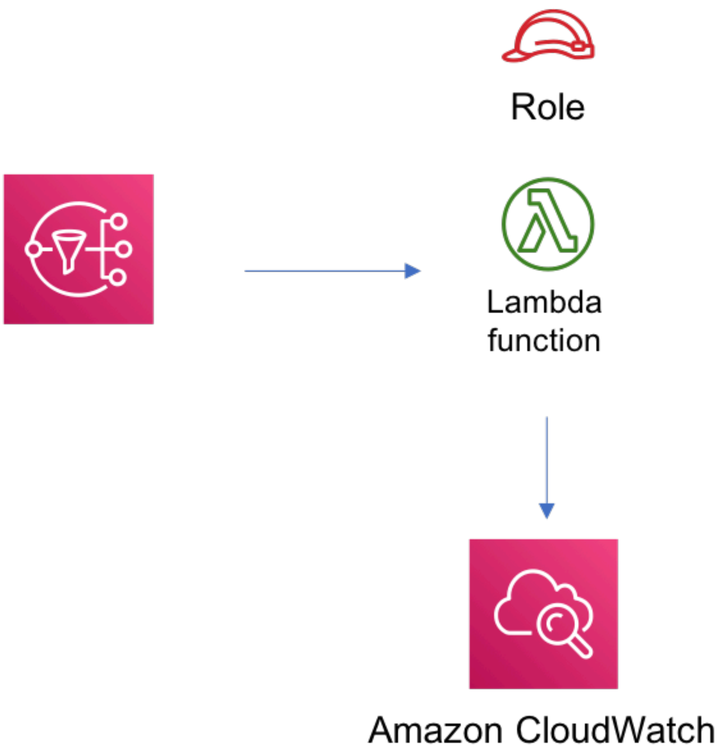
- Configure least privilege access permissions for SNS Topic
- Enable server-side encryption for SNS Topic using AWS managed KMS Key
- Enforce encryption of data in transit

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables

- AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-sns-lambda

aws-sns-sqs

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_sns_sqs
Typescript	@aws-solutions-constructs/aws-sns-sqs
Java	software.amazon.awsconstructs.services.snssqs

Overview

This AWS Solutions Construct implements an Amazon SNS topic connected to an Amazon SQS queue.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { SnsToSqs, SnsToSqsProps } from "@aws-solutions-constructs/aws-sns-sqs";
import * as iam from 'aws-cdk-lib/aws-iam';

const snsToSqsStack = new SnsToSqs(this, 'SnsToSqsPattern', {});

// Grant yourself permissions to use the Customer Managed KMS Key
const policyStatement = new iam.PolicyStatement({
    actions: ["kms:Encrypt", "kms:Decrypt"],
    effect: iam.Effect.ALLOW,
    principals: [ new iam.AccountRootPrincipal() ],
    resources: [ "*" ]
});

snsToSqsStack.encryptionKey?.addToResourcePolicy(policyStatement);
```

Python

```
from aws_solutions_constructs.aws_sns_sqs import SnsToSqs
from aws_cdk import (
    aws_iam as iam,
    Stack
)
from constructs import Construct

construct_stack = SnsToSqs(self, 'SnsToSqsPattern')

policy_statement = iam.PolicyStatement(
    actions=["kms:Encrypt", "kms:Decrypt"],
    effect=iam.Effect.ALLOW,
    principals=[iam.AccountRootPrincipal()],
    resources=["*"]
)

construct_stack.encryption_key.add_to_resource_policy(policy_statement)
```

Java

```
import software.constructs.Construct;
import java.util.List;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.iam.*;
import software.amazon.awsconstructs.services.snssqs.*;

final SnsToSqs constructStack = new SnsToSqs(this, "SnsToSqsPattern",
        new SnsToSqsProps.Builder()
                .build());

// Grant yourself permissions to use the Customer Managed KMS Key
final PolicyStatement policyStatement = PolicyStatement.Builder.create()
        .actions(List.of("kms:Encrypt", "kms:Decrypt"))
        .effect(Effect.ALLOW)
        .principals(List.of(new AccountRootPrincipal()))
        .resources(List.of("*"))
        .build();
```

```
constructStack.getEncryptionKey().addToResourcePolicy(policyStatement);
```

Pattern Construct Props

Name	Type	Description
existingTopicObj?	`sns.Topic`	An optional, existing SNS topic to be used instead of the default topic. Providing both this and `topicProps` will cause an error.
topicProps?	`sns.TopicProps`	Optional user provided properties to override the default properties for the SNS topic.
existingQueueObj?	`sqs.Queue`	An optional, existing SQS queue to be used instead of the default queue. Providing both this and `queueProps` will cause an error.
queueProps?	`sqs.QueueProps`	Optional user provided properties to override the default properties for the SQS queue.
deployDeadLetterQueue?	`boolean`	Whether to create a secondary queue to be used as a dead letter queue. Defaults to true.
deadLetterQueueProps?	`sqs.QueueProps`	Optional user-provided props to override the default props for the dead letter SQS queue.

Name	Type	Description
maxReceiveCount?	`number`	The number of times a message can be unsuccessfully dequeued before being moved to the dead letter queue. Defaults to 15.
enableEncryptionWithCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the queue is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined: topicProps.masterKey, queueProps.encryptionMasterKey, encryptionKey or encryptionKeyProps.
encryptionKey?	`kms.Key`	An optional, imported encryption key to encrypt the SQS Queue and SNS Topic with.
encryptionKeyProps?	`kms.KeyProps`	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SQS queue with.
sqsSubscriptionProps?	`subscriptions.SqsSubscriptionProps`	Optional user-provided props to override the default props for sqsSubscriptionProps.

Pattern Properties

Name	Type	Description
snsTopic	`sns.Topic`	Returns an instance of the SNS topic created by the pattern.
encryptionKey	`kms.Key`	Returns an instance of kms.Key used for the SQS queue, and SNS Topic.
sqsQueue	`sqs.Queue`	Returns an instance of the SQS queue created by the pattern.
deadLetterQueue?	`sqs.Queue`	Returns an instance of the dead-letter SQS queue created by the pattern.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

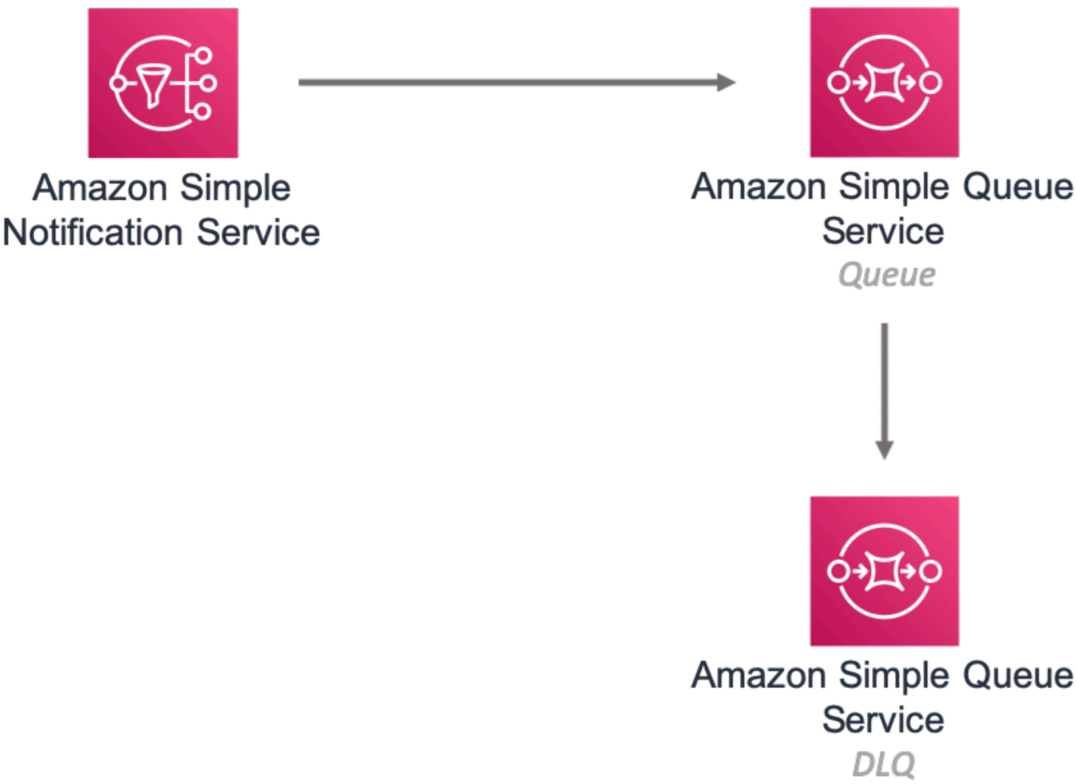
Amazon SNS Topic

- Configure least privilege access permissions for SNS Topic
- Enable server-side encryption for SNS Topic using Customer managed KMS Key
- Enforce encryption of data in transit

Amazon SQS Queue

- Configure least privilege access permissions for SQS Queue
- Deploy SQS dead-letter queue for the source SQS Queue
- Enable server-side encryption for SQS Queue using Customer managed KMS Key
- Enforce encryption of data in transit

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-sns-sqs

aws-sqs-lambda

CFN-RESOURCES STABLE

Language	Package
Python	`aws_solutions_constructs.aws_sqs_lambda`

Language	Package
Typescript	`@aws-solutions-constructs/aws-sqs-lambda`
Java	`software.amazon.awsconstructs.services.sqslambda`

Overview

This AWS Solutions Construct implements an Amazon SQS queue connected to an AWS Lambda function.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { SqsToLambda, SqsToLambdaProps } from "@aws-solutions-constructs/aws-sqs-lambda";
import * as lambda from 'aws-cdk-lib/aws-lambda';

new SqsToLambda(this, 'SqsToLambdaPattern', {
  lambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler',
    code: lambda.Code.fromAsset(`lambda`)
  }
});
```

Python

```
from aws_solutions_constructs.aws_sqs_lambda import SqsToLambda
from aws_cdk import (
    aws_lambda as _lambda,
    Stack
)
```

```
from constructs import Construct


SqsToLambda(self, 'SqsToLambdaPattern',
            lambda_function_props=_lambda.FunctionProps(
                code=_lambda.Code.from_asset('lambda'),
                runtime=_lambda.Runtime.PYTHON_3_9,
                handler='index.handler'
            )
            )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.sqslambda.*;

new SqsToLambda(this, "SnsToSqsPattern", new SqsToLambdaProps.Builder()
        .lambdaFunctionProps(new FunctionProps.Builder()
                .runtime(Runtime.NODEJS_16_X)
                .code(Code.fromAsset("lambda"))
                .handler("index.handler")
                .build())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLambdaObj?	lambda.Function	Existing instance of Lambda Function object, providing both this and lambdaFunctionProps will cause an error.

Name	Type	Description
lambdaFunctionProps?	`lambda.FunctionProps`	User provided props to override the default props for the Lambda function.
existingQueueObj?	`sqs.Queue`	An optional, existing SQS queue to be used instead of the default queue. Providing both this and `queueProps` will cause an error.
queueProps?	`sqs.QueueProps`	Optional user-provided props to override the default props for the SQS queue.
deadLetterQueueProps?	`sqs.QueueProps`	Optional user-provided props to override the default props for the dead letter SQS queue.
deployDeadLetterQueue?	`boolean`	Whether to create a secondary queue to be used as a dead letter queue. Defaults to true.
maxReceiveCount?	`number`	The number of times a message can be unsuccessfully dequeued before being moved to the dead letter queue. Defaults to 15.
sqsEventSourceProps?	`SqsEventSourceProps`	Optional user provided properties for the queue event source.

Name	Type	Description
enableEncryptionWithCustomerManagedKey?	`boolean`	If no key is provided, this flag determines whether the queue is encrypted with a new CMK or an AWS managed key. This flag is ignored if any of the following are defined: queueProps.encryptionMasterKey, encryptionKey or encryptionKeyProps.
encryptionKey?	`kms.Key`	An optional, imported encryption key to encrypt the SQS Queue with.
encryptionKeyProps?	`kms.KeyProps`	Optional user provided properties to override the default properties for the KMS encryption key used to encrypt the SQS queue with.

Pattern Properties

Name	Type	Description
lambdaFunction	`lambda.Function`	Returns an instance of the Lambda function created by the pattern.
sqsQueue	`sqs.Queue`	Returns an instance of the SQS queue created by the pattern.
deadLetterQueue?	`sqs.Queue`	Returns an instance of the dead-letter SQS queue created by the pattern.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

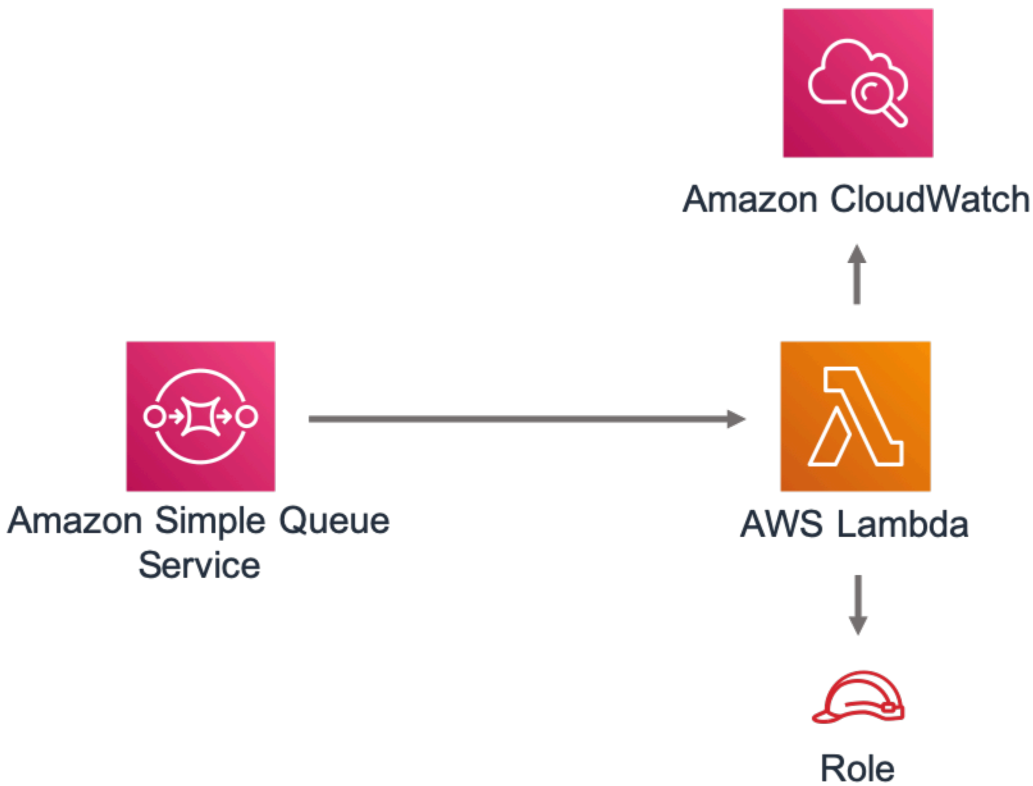
Amazon SQS Queue

- Deploy SQS dead-letter queue for the source SQS Queue
- Enable server-side encryption for source SQS Queue using AWS Managed KMS Key
- Enforce encryption of data in transit

AWS Lambda Function

- Configure limited privilege access IAM role for Lambda function
- Enable reusing connections with Keep-Alive for NodeJs Lambda function
- Enable X-Ray Tracing
- Set Environment Variables
  - AWS_NODEJS_CONNECTION_REUSE_ENABLED (for Node 10.x and higher functions)

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-sqs-lambda

aws-wafwebacl-alb

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_wafwebacl_alb
Typescript	@aws-solutions-constructs/aws-wafwebacl-alb
Java	software.amazon.awsconstructs.services.wafwebaclalb

Overview

This AWS Solutions Construct implements an AWS WAF web ACL connected to an Application Load Balancer.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { WafwebaclToAlbProps, WafwebaclToAlb } from "@aws-solutions-constructs/aws-wafwebacl-alb";

// Use an existing ALB, such as one created by Route53toAlb or AlbToLambda
const existingLoadBalancer = previouslyCreatedLoadBalancer


// Note - all alb constructs turn on ELB logging by default, so require that an
 environment including account
// and region be provided when creating the stack
//
// new MyStack(app, 'id', {env: {account: '123456789012', region: 'us-east-1' }});
//
// This construct can only be attached to a configured Application Load Balancer.
new WafwebaclToAlb(this, 'test-wafwebacl-alb', {
    existingLoadBalancerObj: existingLoadBalancer
});
```

Python

```
from aws_solutions_constructs.aws_route53_alb import Route53ToAlb
from aws_solutions_constructs.aws_wafwebacl_alb import WafwebaclToAlbProps,
 WafwebaclToAlb
from aws_cdk import (
    aws_route53 as route53,
    Stack
)
from constructs import Construct

# Use an existing ALB, such as one created by Route53toAlb or AlbToLambda
existingLoadBalancer = previouslyCreatedLoadBalancer

# Note - all alb constructs turn on ELB logging by default, so require that an
 environment including account
# and region be provided when creating the stack
#
# MyStack(app, 'id', env=cdk.Environment(account='123456789012', region='us-
east-1'))
#
# This construct can only be attached to a configured Application Load Balancer.
WafwebaclToAlb(self, 'test_wafwebacl_alb',
                existing_load_balancer_obj=existingLoadBalancer
                )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.route53.*;
import software.amazon.awsconstructs.services.wafwebaclalb.*;

// Use an existing ALB, such as one created by Route53toAlb or AlbToLambda
final existingLoadBalancer = previouslyCreatedLoadBalancer

// Note - all alb constructs turn on ELB logging by default, so require that an
 environment including account
// and region be provided when creating the stack
```

```
//
// new MyStack(app, "id", StackProps.builder()
//         .env(Environment.builder()
//                 .account("123456789012")
//                 .region("us-east-1")
//                 .build());
//
// This construct can only be attached to a configured Application Load Balancer.
new WafwebaclToAlb(this, "test-wafwebacl-alb", new WafwebaclToAlbProps.Builder()
        .existingLoadBalancerObj(existingLoadBalancer)
        .build());
```

Pattern Construct Props

Name	Type	Description
existingLoadBalancerObj	`elbv2.ApplicationLoadBalancer`	The existing Application Load Balancer Object that will be protected with the WAF web ACL. *Note that a WAF web ACL can only be added to a configured Application Load Balancer, so this construct only accepts an existing ApplicationLoadBalancer and does not accept applicationLoadBalancerProps.*
existingWebaclObj?	`waf.CfnWebACL`	Existing instance of a WAF web ACL, an error will occur if this and props is set.
webaclProps?	`waf.CfnWebACLProps`	Optional user-provided props to override the default props for the AWS WAF web ACL. To use a different collection of managed rule sets, specify a new rules property. Use our `wrapManagedRuleSet(managedG`

Name	Type	Description
		roupName: string, vendorName: string, priority: number) function from core to create an array entry from each desired managed rule set.

Pattern Properties

Name	Type	Description
webacl	waf.CfnWebACL	Returns an instance of the waf.CfnWebACL created by the construct.
loadBalancer	elbv2.ApplicationLoadBalancer	Returns an instance of the Application Load Balancer Object created by the pattern.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS WAF

- Deploy a WAF web ACL with 7 AWS managed rule groups.
  - AWSManagedRulesBotControlRuleSet
  - AWSManagedRulesKnownBadInputsRuleSet
  - AWSManagedRulesCommonRuleSet
  - AWSManagedRulesAnonymousIpList
  - AWSManagedRulesAmazonIpReputationList
  - AWSManagedRulesAdminProtectionRuleSet
  - AWSManagedRulesSQLiRuleSet

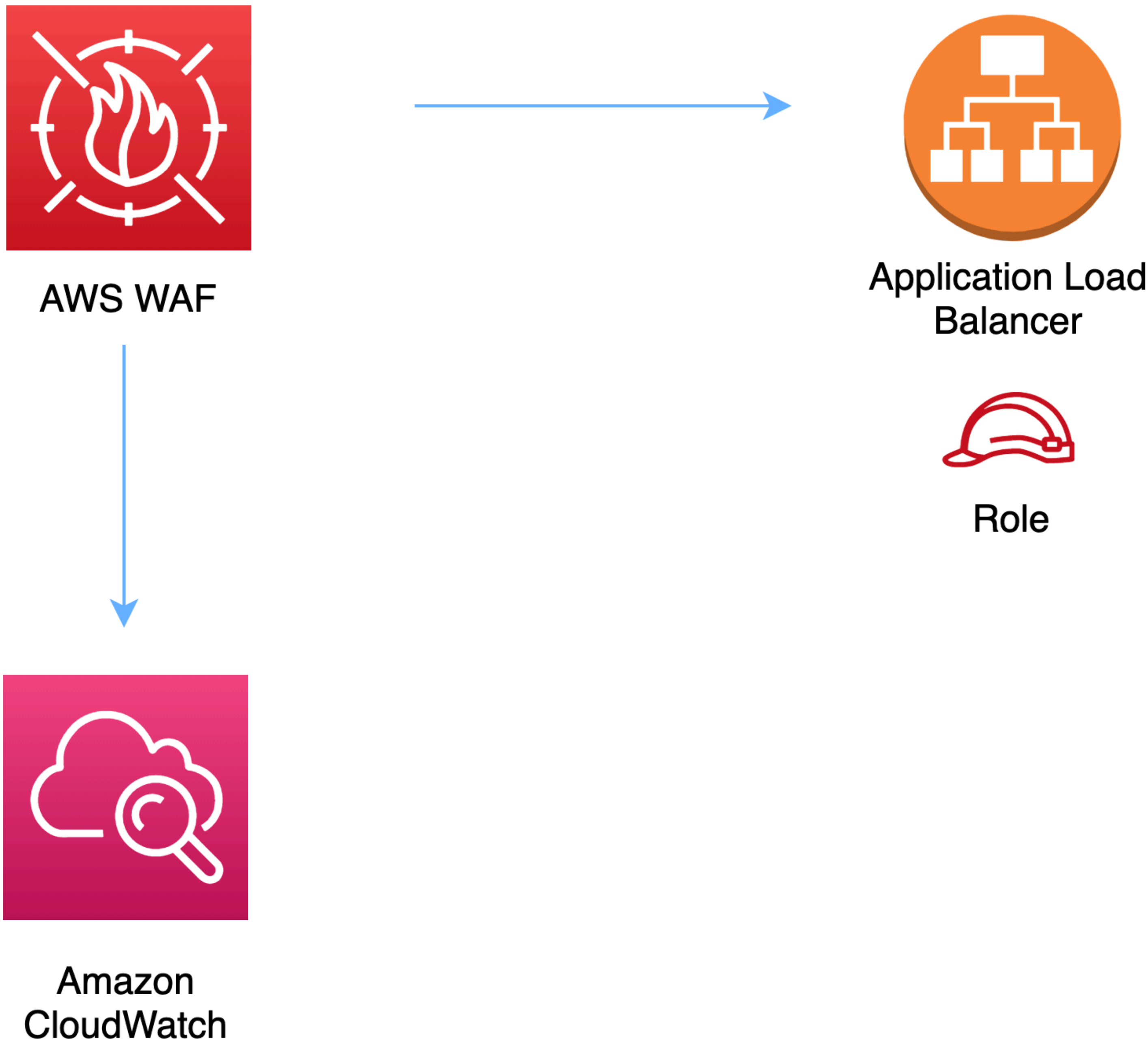
Note that the default rules can be replaced by specifying the rules property of CfnWebACLProps

- Send metrics to Amazon CloudWatch

Application Load Balancer

- User provided Application Load Balancer object is used as-is

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-wafwebacl-alb

aws-wafwebacl-apigateway

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	`aws_solutions_constructs.aws_wafwebacl_apigateway`
Typescript	`@aws-solutions-constructs/aws-wafwebacl-apigateway`
Java	`software.amazon.awsconstructs.services.wafwebaclapigateway`

Overview

This AWS Solutions Construct implements an AWS WAF web ACL connected to Amazon API Gateway REST API.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import * as lambda from "aws-cdk-lib/aws-lambda";
import { ApiGatewayToLambda } from '@aws-solutions-constructs/aws-apigateway-
lambda';
import { WafwebaclToApiGatewayProps, WafwebaclToApiGateway } from "@aws-solutions-
constructs/aws-wafwebacl-apigateway";

const apiGatewayToLambda = new ApiGatewayToLambda(this, 'ApiGatewayToLambdaPattern',
 {
  lambdaFunctionProps: {
    runtime: lambda.Runtime.NODEJS_16_X,
    handler: 'index.handler',
    code: lambda.Code.fromAsset(`lambda`)
  }
});

// This construct can only be attached to a configured API Gateway.
new WafwebaclToApiGateway(this, 'test-wafwebacl-apigateway', {
  existingApiGatewayInterface: apiGatewayToLambda.apiGateway
});
```

Python

```
from aws_solutions_constructs.aws_apigateway_lambda import ApiGatewayToLambda
from aws_solutions_constructs.aws_wafwebacl_apigateway import
 WafwebaclToApiGatewayProps, WafwebaclToApiGateway
from aws_cdk import (
    aws_apigateway as api,
    aws_lambda as _lambda,
    Stack
)
from constructs import Construct

api_gateway_to_lambda = ApiGatewayToLambda(self, 'ApiGatewayToLambdaPattern',
                                    lambda_function_props=_lambda.FunctionProps(
                                        code=_lambda.Code.from_asset(
                                            'lambda'),
                                        runtime=_lambda.Runtime.PYTHON_3_9,
                                        handler='index.handler'
                                    )
                                    )
```

```
# This construct can only be attached to a configured API Gateway.
WafwebaclToApiGateway(self, 'test_wafwebacl_apigateway',
                    existing_api_gateway_interface=api_gateway_to_lambda.api_gateway
                    )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awscdk.services.apigateway.*;
import software.amazon.awscdk.services.lambda.*;
import software.amazon.awscdk.services.lambda.Runtime;
import software.amazon.awsconstructs.services.apigatewaylambda.*;
import software.amazon.awsconstructs.services.wafwebaclapigateway.*;

final ApiGatewayToLambda apiGatewayToLambda = new ApiGatewayToLambda(this,
 "ApiGatewayToLambdaPattern",
        new ApiGatewayToLambdaProps.Builder()
                .lambdaFunctionProps(new FunctionProps.Builder()
                        .runtime(Runtime.NODEJS_16_X)
                        .code(Code.fromAsset("lambda"))
                        .handler("index.handler")
                        .build())
                .build());

// This construct can only be attached to a configured Application Load
// Balancer.
new WafwebaclToApiGateway(this, "test-wafwebacl-apigateway", new
 WafwebaclToApiGatewayProps.Builder()
        .existingApiGatewayInterface(apiGatewayToLambda.getApiGateway())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingApiGateway Interface	[api.IRestApi](#)	The existing API Gateway instance that will be protected with the WAF web ACL. *Note that a WAF web ACL can only be added to a configured API Gateway, so this construct only accepts an existing IRestApi and does not accept apiGatewayProps.*
existingWebaclObj?	[waf.CfnWebACL](#)	Existing instance of a WAF web ACL, an error will occur if this and props is set.
webaclProps?	[waf.CfnWebACLProps](#)	Optional user-provided props to override the default props for the AWS WAF web ACL. To use a different collection of managed rule sets, specify a new rules property. Use our [`wrapManagedRuleSet(managedGroupName: string, vendorName: string, priority: number)`](#) function from core to create an array entry from each desired managed rule set.

Pattern Properties

Name	Type	Description
webacl	`waf.CfnWebACL`	Returns an instance of the waf.CfnWebACL created by the construct.
apiGateway	`api.IRestApi`	Returns an instance of the API Gateway REST API created by the pattern.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS WAF

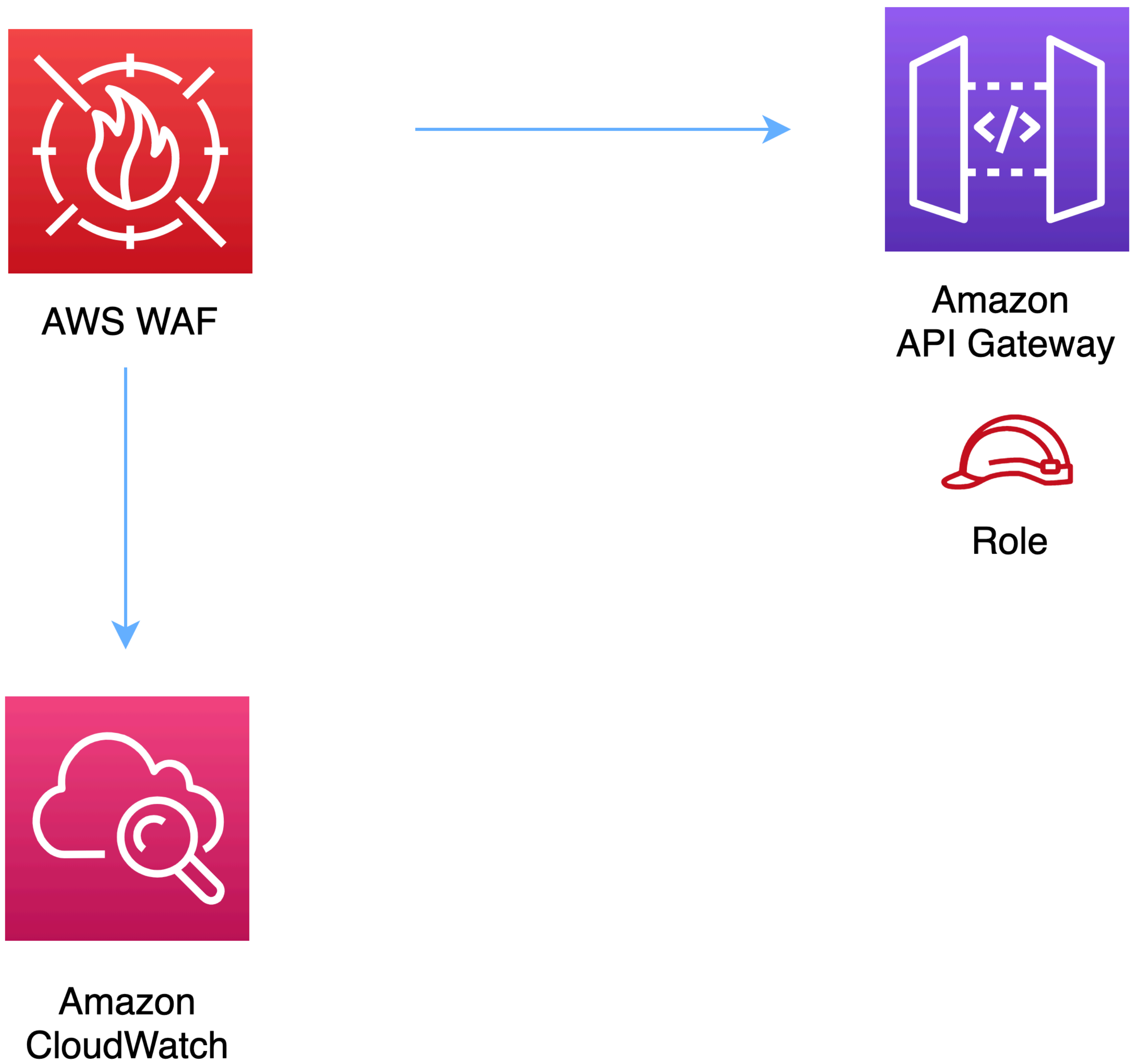
- Deploy a WAF web ACL with 7 AWS managed rule groups.
  - AWSManagedRulesBotControlRuleSet
  - AWSManagedRulesKnownBadInputsRuleSet
  - AWSManagedRulesCommonRuleSet
  - AWSManagedRulesAnonymousIpList
  - AWSManagedRulesAmazonIpReputationList
  - AWSManagedRulesAdminProtectionRuleSet
  - AWSManagedRulesSQLiRuleSet

  Note that the default rules can be replaced by specifying the rules property of CfnWebACLProps
- Send metrics to Amazon CloudWatch

Amazon API Gateway

- User provided API Gateway object is used as-is

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-wafwebacl-apigateway

aws-wafwebacl-appsync

STABILITY EXPERIMENTAL

> All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_wafwebacl_appsync
Typescript	@aws-solutions-constructs/aws-wafwebacl-appsync
Java	software.amazon.awsconstructs.services.wafwebaclappsync

Overview

This AWS Solutions Construct implements an AWS WAF web ACL connected to an AWS AppSync API.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from "constructs";
import { Stack, StackProps } from "aws-cdk-lib";
import {
  WafwebaclToAppsyncProps,
  WafwebaclToAppsync,
} from "@aws-solutions-constructs/aws-wafwebacl-appsync";
```

```
// Use an existing AppSync GraphQL API
const existingGraphQLApi = previouslyCreatedApi;

// This construct can only be attached to a configured AWS AppSync API.
new WafwebaclToAppsync(this, "test-wafwebacl-appsync", {
  existingAppsyncApi: existingGraphQLApi,
});
```

Python

```
from aws_solutions_constructs.aws_wafwebacl_appsync import WafwebaclToAppsyncProps,
 WafwebaclToAppsync
from aws_cdk import (
    aws_route53 as route53,
    Stack
)
from constructs import Construct

# Use an existing AppSync API
existingGraphQLApi = previouslyCreatedApi


# This construct can only be attached to a configured AWS AppSync API.
WafwebaclToAppsync(self, 'test_wafwebacl_appsync',
                existing_appsync_api=existingGraphQLApi
                )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.wafwebaclappsync.*;

// Use an existing AppSync API
final existingGraphQLApi = previouslyCreatedApi


// This construct can only be attached to a configured AWS AppSync API.
```

```
new WafwebaclToAppsync(this, "test-wafwebacl-appsync", new
 WafwebaclToAppsyncProps.Builder()
        .existingAppsyncApi(existingGraphQLApi)
        .build());
```

Pattern Construct Props

N	Type	Description
ex pr	appsync.CfnGraphQLApi	The existing Appsync CfnGraphQLApi object that will be protected with the WAF web ACL. *Note that a WAF web ACL can only be added to a configured AppSync API, so this construct only accepts an existing CfnGraphQLApi and does not accept CfnGraphQLApiProps.*
ex eb	waf.CfnWebACL	Existing instance of a WAF web ACL, an error will occur if this and props is set.
w ps	waf.CfnWebACLProps	Optional user-provided props to override the default props for the AWS WAF web ACL. To use a different collection of managed rule sets, specify a new rules property. Use our `wrapManagedRuleSet(managedGroupName: string, vendorName: string, priority: number)` function from core to create an array entry from each desired managed rule set.

Pattern Properties

Nar	Type	Description
wek	waf.CfnWebACL	Returns an instance of the waf.CfnWebACL created by the construct.
app i	appsync.CfnGraphQLApi	Returns an instance of the CfnGraphQLApi used by the pattern.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS WAF

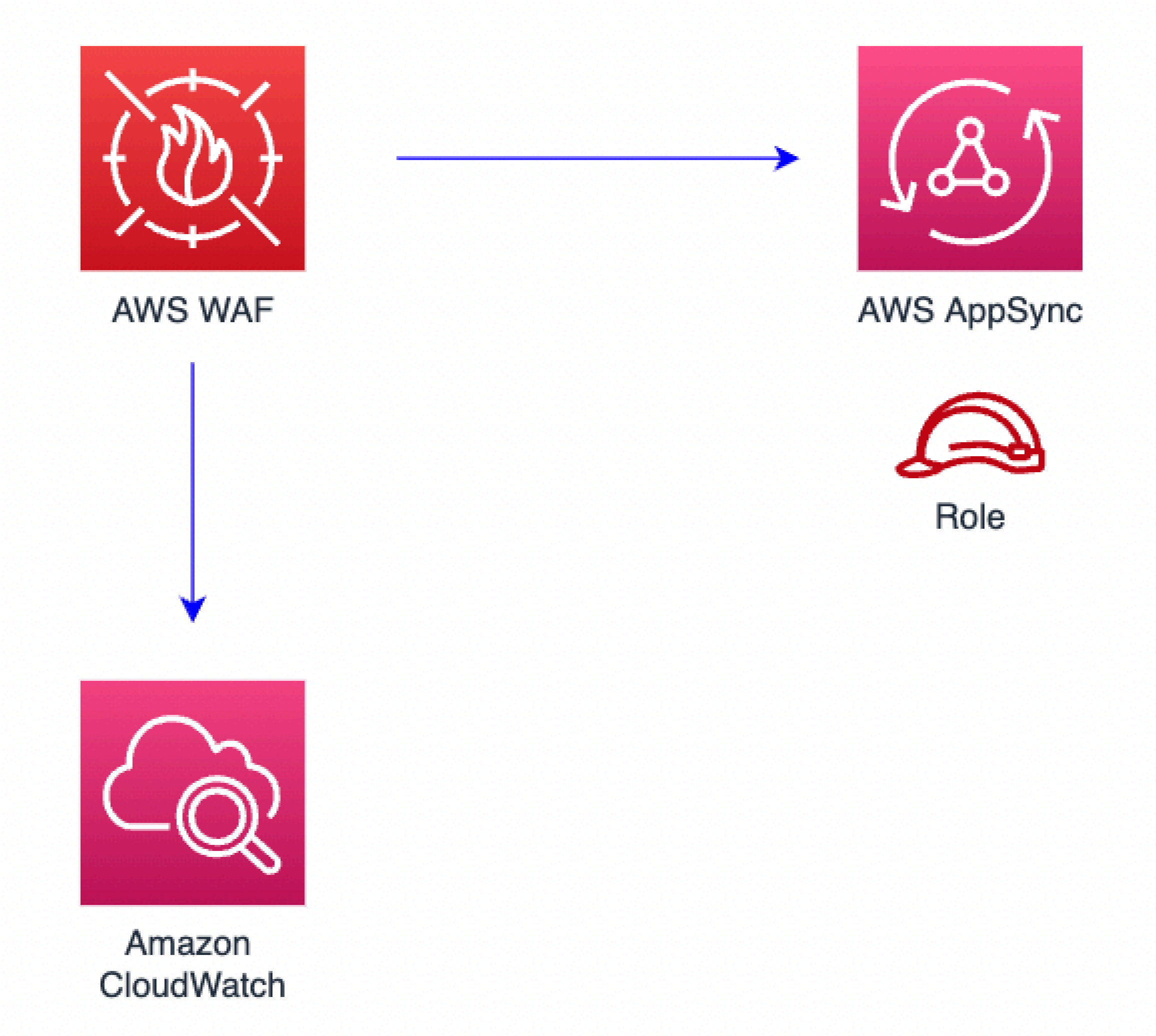
- Deploy a WAF web ACL with 7 AWS managed rule groups.
  - AWSManagedRulesBotControlRuleSet
  - AWSManagedRulesKnownBadInputsRuleSet
  - AWSManagedRulesCommonRuleSet
  - AWSManagedRulesAnonymousIpList
  - AWSManagedRulesAmazonIpReputationList
  - AWSManagedRulesAdminProtectionRuleSet
  - AWSManagedRulesSQLiRuleSet

  Note that the default rules can be replaced by specifying the rules property of CfnWebACLProps
- Send metrics to Amazon CloudWatch

AppSync API

- User provided AppSync graphql API object is used as-is

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-wafwebacl-appsync

aws-wafwebacl-cloudfront

STABILITY EXPERIMENTAL

All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

Language	Package
Python	aws_solutions_constructs.aws_wafwebacl_cloudfront
Typescript	@aws-solutions-constructs/aws-wafwebacl-cloudfront
Java	software.amazon.awsconstructs.services.wafwebaclcloudfront

Overview

This AWS Solutions Construct implements an AWS WAF web ACL connected to Amazon CloudFront.

Here is a minimal deployable pattern definition:

Typescript

```
import { Construct } from 'constructs';
import { Stack, StackProps } from 'aws-cdk-lib';
import { CloudFrontToS3 } from '@aws-solutions-constructs/aws-cloudfront-s3';
import { WafwebaclToCloudFront } from "@aws-solutions-constructs/aws-wafwebacl-
cloudfront";

const cloudfrontToS3 = new CloudFrontToS3(this, 'test-cloudfront-s3', {});

// This construct can only be attached to a configured CloudFront.
new WafwebaclToCloudFront(this, 'test-wafwebacl-cloudfront', {
    existingCloudFrontWebDistribution: cloudfrontToS3.cloudFrontWebDistribution
});
```

Python

```
from aws_solutions_constructs.aws_cloudfront_s3 import CloudFrontToS3
from aws_solutions_constructs.aws_wafwebacl_cloudfront import WafwebaclToCloudFront
from aws_cdk import Stack
from constructs import Construct

cloudfront_to_s3 = CloudFrontToS3(self, 'test_cloudfront_s3')

# This construct can only be attached to a configured CloudFront.
WafwebaclToCloudFront(self, 'test_wafwebacl_cloudfront',

 existing_cloud_front_web_distribution=cloudfront_to_s3.cloud_front_web_distribution
                      )
```

Java

```
import software.constructs.Construct;

import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;
import software.amazon.awsconstructs.services.cloudfronts3.*;
import software.amazon.awsconstructs.services.wafwebaclcloudfront.*;

final CloudFrontToS3 cloudfrontToS3 = new CloudFrontToS3(this, "test-cloudfront-s3",
        new CloudFrontToS3Props.Builder()
                .build());

// This construct can only be attached to a configured CloudFront.
new WafwebaclToCloudFront(this, "test-wafwebacl-cloudfront", new
 WafwebaclToCloudFrontProps.Builder()

 .existingCloudFrontWebDistribution(cloudfrontToS3.getCloudFrontWebDistribution())
        .build());
```

Pattern Construct Props

Name	Type	Description
existingCloudFront WebDistribution	`cloudfront.Distribution`	The existing CloudFront instance that will be protected with the WAF web ACL. *Note that a WAF web ACL can only be added to a configured CloudFront, so this construct only accepts an existing Distribution and does not accept cloudfrontProps.*
existingWebaclObj?	`waf.CfnWebACL`	Existing instance of a WAF web ACL, an error will occur if this and props is set.
webaclProps?	`waf.CfnWebACLProps`	Optional user-provided props to override the default props for the AWS WAF web ACL. To use a different collection of managed rule sets, specify a new rules property. Use our `wrapManagedRuleSet(managedGroupName: string, vendorName: string, priority: number)` function from core to create an array entry from each desired managed rule set.

Pattern Properties

Name	Type	Description
webacl	waf.CfnWebACL	Returns an instance of the waf.CfnWebACL created by the construct.
cloudFrontWebDistribution	cloudfront.Distribution	Returns an instance of cloudfront.Distribution created by the construct.

Default settings

Out of the box implementation of the Construct without any override will set the following defaults:

AWS WAF

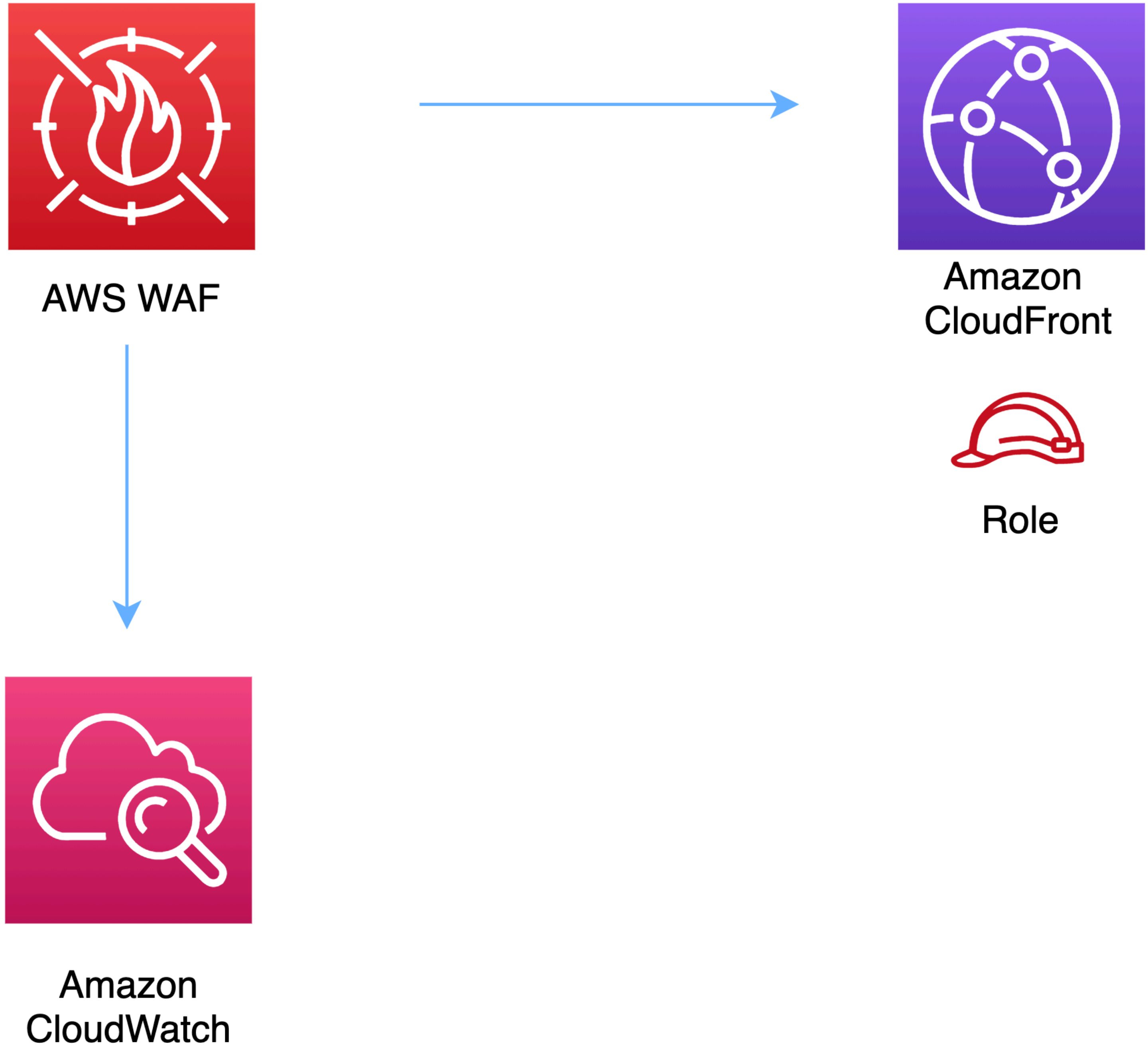
- Deploy a WAF web ACL with 7 AWS managed rule groups.
  - AWSManagedRulesBotControlRuleSet
  - AWSManagedRulesKnownBadInputsRuleSet
  - AWSManagedRulesCommonRuleSet
  - AWSManagedRulesAnonymousIpList
  - AWSManagedRulesAmazonIpReputationList
  - AWSManagedRulesAdminProtectionRuleSet
  - AWSManagedRulesSQLiRuleSet

  Note that the default rules can be replaced by specifying the rules property of CfnWebACLProps
- Send metrics to Amazon CloudWatch

Amazon CloudFront

- User provided CloudFront object is used as-is

Architecture



GitHub

To view the code for this pattern, create/view issues and pull requests, and more:	
	@aws-solutions-constructs/aws-wafwebacl-cloudfront

core

STABILITY EXPERIMENTAL

All classes are under active development and subject to non-backward compatible changes or removal in any future version. These are not subject to the Semantic Versioning model. This means that while you may use them, you may need to update your source code when upgrading to a newer version of this package.

The core library includes the basic building blocks of AWS Solutions Constructs. It defines the core classes that are used in the rest of AWS Solutions Constructs.

Default Properties for AWS CDK Constructs

Core library sets the default properties for the AWS CDK Constructs used by AWS Solutions Constructs constructs.

For example, the following is the snippet of default properties for S3 Bucket construct created by AWS Solutions Constructs construct. By default, it will turn on the server-side encryption, bucket versioning, block all public access and setup the S3 access logging.

```
{
  encryption: s3.BucketEncryption.S3_MANAGED,
  versioned: true,
  blockPublicAccess: s3.BlockPublicAccess.BLOCK_ALL,
  removalPolicy: RemovalPolicy.RETAIN,
  serverAccessLogsBucket: loggingBucket
}
```

Override the default properties

The default properties set by the Core library can be overridden by user provided properties. For example, the user can override the Amazon S3 Block Public Access property to meet specific requirements.

```
  const stack = new cdk.Stack();

  const props: CloudFrontToS3Props = {
```

```
    bucketProps: {
      blockPublicAccess: {
        blockPublicAcls: false,
        blockPublicPolicy: true,
        ignorePublicAcls: false,
        restrictPublicBuckets: true
      }
    }
  };

  new CloudFrontToS3(stack, 'test-cloudfront-s3', props);

  expect(stack).toHaveResource("AWS::S3::Bucket", {
    PublicAccessBlockConfiguration: {
      BlockPublicAcls: false,
      BlockPublicPolicy: true,
      IgnorePublicAcls: false,
      RestrictPublicBuckets: true
    },
  });
```

Property override warnings

When a default property from the Core library is overridden by a user-provided property, Constructs will emit one or more warning messages to the console highlighting the change(s). These messages are intended to provide situational awareness to the user and prevent unintentional overrides that could create security risks. These messages will appear whenever deployment/build-related commands are executed, including `cdk deploy`, `cdk synth`, `npm test`, etc.

Example message: `AWS_CONSTRUCTS_WARNING: An override has been provided for the property: BillingMode. Default value: 'PAY_PER_REQUEST'. You provided: 'PROVISIONED'.`

Toggling override warnings

Override warning messages are enabled by default, but can be explicitly turned on/off using the `overrideWarningsEnabled` shell variable.

- To explicitly *turn off* override warnings, run `export overrideWarningsEnabled=false`.
- To explicitly *turn on* override warnings, run `export overrideWarningsEnabled=true`.

- To revert to the default, run `unset overrideWarningsEnabled`.

Document Revisions

To be notified about updates to AWS Solutions Constructs, subscribe to the RSS feed.

Change	Description	Date
Content updated	Updated documents to reflect version 2.56.0. Added new page for `aws-constructs-fa ctories`. (v2.56.0)	April 29, 2024
Content updated	Updated documents to reflect version 2.51.0 (v2.51.0)	February 5, 2024
Content updated	Updated documents to reflect version 2.44.0. Added new `aws-lambda-kendra` and `aws-openapigateway-lambda` constructs. (v2.44.0)	October 11, 2023
Content updated	Updated documents to reflect version 2.32.0. (v2.32.0)	February 27, 2023
Content updated	Updated documents to reflect version 2.30.0. Added new `aws-fargate-kinesisfirehos e`, `aws-fargate-kines isstreams`, `aws-lambda-kinesisfirehose`, and `aws-lambda-kinesisstreams` constructs. (v2.30.0)	January 3, 2023
Content updated	Updated documents to reflect version 2.27.0. (v2.27.0)	November 11, 2022
Content updated	Minor props updates for constructs using AWS CloudFront. (v2.14.0)	August 15, 2022

Content updated	Updated properties for the `aws-lambda-elasticsearch-kibana` construct. Minor typographical corrections. (v2.11.0)	July 19, 2022
Content updated	Updated documents to reflect version 2.10.0. (v2.10.0)	July 14, 2022
Content updated	Added minimal deployable code samples for TypeScript, Python, and Java for each Construct. Added `aws-fargate-dynamodb`, `aws-fargate-secretsmanager`, `aws-fargate-ssmstringparameter`, `aws-fargate-stepfunctions`, and `aws-lambda-elasticachememcached` Constructs. (v1.156.0)	May 20, 2022
Content updated	Updated walkthrough content to reflect development with AWS Solutions Constructs v2.	May 12, 2022
Content updated	Added `aws-fargate-s3` and `aws-fargate-sqs` constructs.	March 7, 2022
Content updated	Added `aws-fargate-sns` and `aws-route53-apigateway` constructs. Updated properties for `aws-alb-fargate` constructs.	February 1, 2022

Content updated	Added `aws-alb-fargate` and `aws-iot-s3` constructs. Updated Node.js runtime specifications in minimal deployable code snippets for applicable patterns. Updated logging properties for constructs that deploy AWS Step Functions state machines. Other minor updates.	January 14, 2022
Content updated	Added `logS3AccessLogs` and `loggingBucketProps` input properties, as well as an `S3BucketInterface` output property, to applicable patterns that deploy an Amazon S3 bucket. Other minor updates. (v2.0.0)	December 3, 2021
Content updated	Added new `aws-alb-lambda`, `aws-route53-alb`, and `aws-wafwebacl-alb` constructs. Promoted additional constructs to stable from experimental. Other minor updates. (v1.126.0)	October 27, 2021

Content updated	Added new `loggingBucketProps` property to the following patterns: `aws-cloudfront-s3` , `aws-lambda-s3` , `aws-s3-lambda` , `aws-s3-sqs` , `aws-s3-step-function` , and `aws-s3-stepfunctions` . (v1.125.0)	October 11, 2021
Content updated	Added new `aws-wafwebacl-cloudfront` pattern. Updated properties for `aws-apigateway-kinesisstreams` pattern. (v1.124.0)	October 3, 2021
Content updated	Added new `aws-iot-kinesisstreams` pattern. Other minor content updates. (v1.123.0)	September 21, 2021
Content updated	Added new `aws-lambda-eventbridge` and `aws-wafwebacl-apigateway` pattern. Other minor content updates. (v1.122.0)	September 20, 2021
Content updated	Added new `aws-iot-sqs` pattern. Other minor content updates. (v1.117.0)	August 17, 2021

Content updated	Multiple patterns deprecate d and replaced with new patterns based on updated naming convention. Multiple patterns upgraded to Stable. Other minor updates for v1.116.0.	August 17, 2021
Content updated	Updated properties for the aws-cloudfront-s3 pattern.	July 26, 2021
Content updated	Updated properties for the aws-cloudfront-s3 pattern. Other minor content updates.	July 23, 2021
Content updated	Updated properties for select patterns and added new use case.	June 16, 2021
Content updated	Added aws-lambda-ssmstri ngparameter pattern. Other minor content updates.	May 27, 2021
Content updated	Added aws-lambda-secrets manager pattern. Other minor content updates.	May 12, 2021
Content updated	Property updates to select *-lambda patterns. Other minor content updates.	April 17, 2021
Content updated	Fixed an issue in the Walkthrough for Python users and updated property examples for constructs containing Lambda functions.	March 30, 2021

Content updated	Minor fixes/updates to pattern props and default settings for select patterns.	March 8, 2021
Content updated	Minor fixes/updates to walkthrough content.	March 4, 2021
Content updated	Added `aws-lambda-sagemakerendpoint` pattern and updated properties for select Kinesis Firehose patterns.	February 24, 2021
Content updated	Added `aws-kinesisstreams-gluejob` pattern and updated walkthrough steps for Python users.	February 17, 2021
Content updated	Updated properties for `aws-cloudfront-*` patterns.	February 9, 2021
Content updated	Added link to GitHub for each pattern.	February 5, 2021
Content updated	Updated properties for select patterns.	February 1, 2021
Content updated	Updated documentation of properties and default settings for select patterns.	January 4, 2021
Content updated	Added new patterns: aws-cloudfront-mediastore and aws-s3-sqs.	December 20, 2020
Content updated	Removed aws-lambda-sagemaker pattern.	November 17, 2020

Content updated	Added new patterns: aws-events-rule-kinesisstreams, aws-events-rule-kinesisfirehose-s3, and aws-lambda-sagemaker.	October 27, 2020
Content updated	Updated to reflect breaking change in aws-events-rule-sns and aws-events-rule-sqs patterns: class and interface names changed to pascal case.	October 22, 2020
Content updated	Added aws-apigateway-sagemakerendpoint and aws-kinesisstreams-kinesisfirehose-s3 patterns; other minor updates to existing content.	October 20, 2020
Content updated	Added aws-apigateway-iot pattern; other minor updates to existing content.	October 7, 2020
Content updated	Updated minimal deployable pattern code snippets and best practice defaults for all patterns.	October 5, 2020
Content updated	Updated properties for aws-kinesisstreams-lambda pattern to reflect breaking change.	September 14, 2020
Content updated	Minor fix to second part of walkthrough.	September 10, 2020

Content updated	Added aws-apigateway-kinesisstreams, aws-events-rule-sns, and aws-events-rule-sqs patterns.	September 10, 2020
Content updated	Added aws-sns-sqs pattern; updates to all SNS patterns; minor typographical corrections.	September 2, 2020
Content updated	Fixed module names for aws-sqs-lambda pattern.	August 31, 2020
Content updated	Fixed Python module name for aws-dynamodb-stream-lambda-elasticsearch-kibana pattern.	August 31, 2020
Content updated	Updated defaults for Lambda patterns; other minor updates.	August 27, 2020
Content updated	Updated public properties for S3 patterns; updated defaults for DynamoDB patterns.	August 10, 2020
Content updated	Updated multiple patterns to highlight default enforcement of encryption in transit.	August 4, 2020
Content updated	Added aws-lambda-sqs-lambda pattern; improved configuration instructions in Getting Started guide; updated all patterns to make additional resources available through public properties.	July 27, 2020

Content updated	Added aws-lambda-sqs pattern; other minor updates.	July 20, 2020
Content updated	Removed deployLambda and deployBucket properties from relevant patterns; other minor updates.	July 9, 2020
Content updated	Added aws-lambda-step-fu nction pattern and corrected minor typographical errors.	July 7, 2020
Content updated	Added existingTableObj? property to select DynamoDB patterns.	June 25, 2020
Content updated	Several text corrections and fixes for broken links.	June 23, 2020
Initial release	AWS Solutions Constructs made publicly available.	June 22, 2020

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.