Soluções da AWS

Constructos da AWS

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

Constructos da AWS: Soluções da AWS

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

As marcas comerciais e o visual comercial da Amazon não podem ser usados em conexão com nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa causar confusão entre os clientes ou que deprecie ou desacredite a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, conectados ou patrocinados pela Amazon.

Table of Contents

Visão geral ... 1
O que é Constructos das Soluções da AWS? ... 1
Por que usar o AWS Solutions Constructs? ... 1
Conceitos básicos ... 3
Pré-requisitos ... 3
Instalar o AWS CDK ... 4
Trabalhar com construções de soluções da AWS ... 4
Passo a passo - Parte 1 ... 5
Hello constructos ... 5
Criando o diretório de aplicativos e inicializando o CDK da AWS ... 6
Atualizar dependências da base do projeto ... 7
Código de manipulador do Lambda ... 9
Instale as dependências do AWS CDK e do AWS Solutions Constructs ... 10
Adicione um padrão do Amazon API Gateway/AWS Lambda à sua pilha ... 12
Implantação cdk ... 18
Saídas da pilha ... 19
Testar seu aplicativo ... 19
Passo a passo - Parte 2 ... 19
Código do Lambda do contador ... 20
Instalar as novas dependências ... 22
Defina os recursos ... 23
Review as alterações ... 36
Implantação cdk ... 37
Saídas da pilha ... 38
Testar seu aplicativo ... 38
Casos de uso do ... 39
Site do AWS Static S3 ... 40
Manipulador de imagens simples sem servidor da AWS ... 40
Aplicativo web sem servidor da AWS ... 40
Referência de API ... 42
Modules ... 42
Conteúdo do Módulo ... 42
aws-apigateway-dynamodb ... 43
Visão geral ... 43

Inicializador ... 44
Adereços de construção de padrão ... 44
Propriedades de padrão ... 46
Configurações padrão ... 46
Arquitetura ... 47
GitHub ... 48
aws-apigateway-muito ... 48
Visão geral ... 48
Inicializador ... 49
Props de construção de padrão ... 49
Propriedades de padrão ... 50
Configurações padrão ... 51
Arquitetura ... 54
Exemplos ... 54
GitHub ... 56
aws-apigateway-kinesisstreams ... 56
Visão geral ... 57
Inicializador ... 57
Adereços de construção de padrão ... 58
Propriedades do padrão ... 59
Uso da API de amostra ... 60
Configurações padrão ... 61
Arquitetura ... 62
GitHub ... 62
aws-apigateway-lambda ... 62
Visão geral ... 63
Inicializador ... 63
Adereços de construção ... 64
Propriedades de padrão ... 65
Configurações padrão ... 65
Arquitetura ... 66
GitHub ... 66
aws-apigateway-sagemakerendpoint ... 66
Visão geral ... 67
Inicializador ... 68
Adereços de construção de padrão ... 68

Propriedades de padrão 70
Uso da API de amostra 60
Configurações padrão 71
Arquitetura 72
GitHub 72
aws-apigateway-sqs 72
Visão geral 73
Inicializador 73
Adereços de construção de padrão 74
Propriedades de padrão 76
Uso da API de amostra 60
Configurações padrão 77
Arquitetura 78
GitHub 78
aws-cloudfront-apigateway 78
Visão geral 79
Inicializador 80
Adereços de construção de padrão 80
Propriedades de padrão 81
Configurações padrão 81
Arquitetura 82
GitHub 83
aws-cloudfront-apigateway-lambda 83
Visão geral 83
Inicializador 84
Adereços de construção de padrão 84
Propriedades de padrão 86
Configurações padrão 87
Arquitetura 88
GitHub 88
aws-cloudfront-mediastore 88
Visão geral 89
Inicializador 89
Props de criação de padrão 90
Propriedades de padrão 90
Configurações padrão 91

Arquitetura 92
GitHub 93
aws-cloudfront-s3 93
Visão geral 93
Inicializador 94
Props de criação de padrão 94
Propriedades do padrão 95
Configurações padrão 96
Arquitetura 97
GitHub 97
aws-cognito-apigateway-lambda 97
Visão geral 79
Inicializador 99
Adereços de construção de padrão 100
Propriedades de padrão 101
Configurações padrão 102
Arquitetura 103
GitHub 103
aws-dynamodb-stream-lambda 103
Visão geral 104
Inicializador 105
Adereços de criação de padrão 105
Propriedades do padrão 106
Função Lambda 106
Configurações padrão 106
Arquitetura 108
GitHub 108
aws-dynamodb-stream-lambda-elasticsearch-kibana 108
Visão geral 109
Inicializador 110
Props de criação de padrão 110
Propriedades do padrão 111
Função Lambda 112
Configurações padrão 112
Arquitetura 114
GitHub 114

aws-events-rule-kinesisfirehose-s3 114
Visão geral 115
Inicializador 116
Estrutura de padrão 116
Propriedades de padrão 117
Configuração padrão 118
Arquitetura 119
GitHub 119
aws-events-rule-kinesisstreams 119
Visão geral 120
Inicializador 121
Estrutura de padrão 121
Propriedades de padrão 122
Configurações padrão 122
Arquitetura 123
GitHub 123
aws-events-rule-lambda 123
Visão geral 124
Inicializador 125
Adereços de criação de padrão 125
Propriedades de padrão 126
Configuração padrão 126
Arquitetura 127
GitHub 127
aws-events-rule-sns 127
Visão geral 128
Inicializador 129
Adereços de construção de padrão 129
Propriedades de padrão 130
Configurações padrão 131
Arquitetura 132
GitHub 132
aws-events-rule-sqs 132
Visão geral 133
Inicializador 134
Adereços de construção de padrão 134

Propriedades de padrão 136
Configurações padrão 137
Arquitetura 138
GitHub 138
aws-events-rule-step-function 138
Visão geral 139
Inicializador 140
Adereços de criação de padrão 140
Propriedades de padrão 141
Configurações padrão 141
Arquitetura 142
GitHub 142
aws-iot-kinesisfirehose-s3 142
Visão geral 143
Inicializador 144
Adereços de criação de padrão 144
Propriedades do padrão 145
Configurações padrão 146
Arquitetura 147
GitHub 147
aws-iot-lambda 147
Visão geral 148
Inicializador 149
Adereços de construção de padrão 149
Propriedades de padrão 150
Configurações padrão 150
Arquitetura 151
GitHub 151
aws-iot-lambda-dynamodb 151
Visão geral 152
Inicializador 153
Props de criação de padrão 153
Propriedades de padrão 154
Configurações padrão 154
Arquitetura 156
GitHub 156

aws-kinesisfirehose-s3 156
Visão geral 157
Inicializador 157
Props de criação de padrão 158
Propriedades do padrão 158
Configurações padrão 159
Arquitetura 160
GitHub 160
aws-kinesisfirehose-s3-e-kinesisanalytics 161
Visão geral 161
Inicializador 162
Padrão de criação 163
Propriedades de padrão 164
Configurações padrão 164
Arquitetura 165
GitHub 166
aws-kinesisstreams-gluejob 166
Visão geral 166
Inicializador 168
Adereços de construção padrão 168
SinkDataStoreProps 170
SinkStoretype 170
Configurações padrão 171
Arquitetura 173
GitHub 173
aws-kinesisstreams-kinesisfirehose-s3 173
Visão geral 174
Inicializador 174
Adereços de criação de padrão 175
Propriedades do padrão 176
Configurações padrão 177
Arquitetura 178
GitHub 178
aws-kinesisstreams-lambda 178
Visão geral 179
Inicializador 180

Props de criação de padrão 180
Propriedades de padrão 181
Configurações padrão 182
Arquitetura 183
GitHub 183
aws-lambda-dynamodb 183
Visão geral 184
Inicializador 185
Aderetos de criação de padrão 185
Propriedades de padrão 188
Configurações padrão 189
Arquitetura 190
GitHub 190
aws-lambda-elasticsearch-kibana 190
Visão geral 191
Inicializador 192
Estrutura de construção de padrão 192
Propriedades do padrão 193
Função Lambda 194
Configurações padrão 194
Arquitetura 196
GitHub 196
aws-lambda-s3 197
Visão geral 197
Inicializador 198
Adereços de construção de padrão 198
Propriedades de padrão 201
Configurações padrão 202
Arquitetura 203
GitHub 203
aws-lambda-ssmstringparameter 203
Visão geral 204
Inicializador 205
Adereços de criação de padrão 205
Propriedades de padrão 210
Configurações padrão 210

Arquitetura ... 211
GitHub ... 211
aws-lambda-sagemakerendpoint ... 211
Visão geral ... 212
Inicializador ... 213
Adereços de criação de padrão ... 213
Propriedades de padrão ... 217
Configurações padrão ... 217
Arquitetura ... 218
GitHub ... 219
aws-lambda-secretsmanager ... 219
Visão geral ... 219
Inicializador ... 220
Adereços de criação de padrão ... 220
Propriedades de padrão ... 223
Configurações padrão ... 224
Arquitetura ... 225
GitHub ... 225
aws-lambda-sns ... 225
Visão geral ... 226
Inicializador ... 226
Adereços de criação de padrão ... 227
Propriedades de padrão ... 231
Configurações padrão ... 231
Arquitetura ... 232
GitHub ... 232
aws-lambda-sqs ... 232
Visão geral ... 233
Inicializador ... 233
Adereços de construção de padrão ... 234
Propriedades de padrão ... 238
Configurações padrão ... 239
Arquitetura ... 240
GitHub ... 240
aws-lambda-sqs-lambda ... 240
Visão geral ... 241

Inicializador ... 242
Adereços de construção de padrão ... 242
Propriedades de padrão ... 244
Configurações padrão ... 245
Arquitetura ... 246
GitHub ... 246
aws-lambda-step-function ... 246
Visão geral ... 247
Inicializador ... 248
Conceitos de criação de padrão ... 248
Propriedades de padrão ... 249
Configuração padrão ... 250
Arquitetura ... 251
GitHub ... 251
aws-s3-lambda ... 251
Visão geral ... 252
Inicializador ... 253
Adereços de construção de padrão ... 253
Propriedades de padrão ... 254
Configurações padrão ... 254
Arquitetura ... 255
GitHub ... 256
aws-s3-sqs ... 256
Visão geral ... 256
Inicializador ... 257
Adereços de criação de padrão ... 257
Propriedades do padrão ... 260
Configurações padrão ... 260
Arquitetura ... 261
GitHub ... 261
aws-s3-step-function ... 261
Visão geral ... 262
Inicializador ... 263
Adereços de construção de padrão ... 263
Propriedades do padrão ... 264
Configurações padrão ... 265

Arquitetura 267
GitHub 267
aws-sns-lambda 267
Visão geral 268
Inicializador 268
Adereços de criação de padrão 269
Propriedades de padrão 270
Configurações padrão 270
Arquitetura 271
GitHub 271
aws-sns-sqs 271
Visão geral 272
Inicializador 273
Adereços de construção de padrão 273
Propriedades de padrão 275
Configurações padrão 275
Arquitetura 276
GitHub 277
aws-sqs-lambda 277
Visão geral 277
Inicializador 278
Adereços de construção de padrão 278
Propriedades de padrão 280
Configurações padrão 280
Arquitetura 281
GitHub 281
core 281
Propriedades padrão para construções CDK da AWS 282
Substituir as propriedades padrão 282
Avisos de substituição de propriedades 283
Revisões do documento 284
Avisos 289
..... ccxc

Constructos das soluções da AWS

Data de publicação: Maio de 2021(Revisões do documento)

O que é Constructos das Soluções da AWS?

AWS Solutions Constructs (Constructs) é uma extensão de código aberto doKit de desenvolvimento da Nuvem AWS (AWS CDK)que fornece padrões multi-serviço e bem arquitetados para definir rapidamente soluções em código para criar infraestrutura previsível e repetível. O objetivo é acelerar a experiência para os desenvolvedores criarem soluções de qualquer tamanho usando definições baseadas em padrões para sua arquitetura.

Use o AWS Solutions Constructs para definir suas soluções em uma linguagem de programação familiar. As construções de soluções da AWS oferecem suporte a TypeScript, JavaScript, Python e Java no momento.

Para navegar pelo catálogo completo de padrões de construções de soluções da AWS,Clique aqui.

Por que usar o AWS Solutions Constructs?

Com a taxa de inovação dos provedores de nuvem, conhecer e entender as melhores práticas e garantir que elas sejam implementadas corretamente em sua solução pode ser assustador. Constructs permite combinar padrões pré-construídos e bem arquitetados e casos de uso que executam ações comuns usando serviços de nuvem de forma escalável e segura. Como o Constructs fornece uma biblioteca para linguagens de programação modernas, você pode aplicar habilidades de desenvolvimento existentes e ferramentas familiares à tarefa de criar uma infraestrutura de nuvem bem arquitetada para suas soluções.

Outras vantagens dos Construtos de soluções da AWS incluem:

- Ele é criado com base na estrutura de desenvolvimento de software livre do Kit de Desenvolvimento da Nuvem AWS (AWS CDK).
- Use lógica (if instruções, for-loops etc.) ao definir sua infra-estrutura de solução.
- Use técnicas orientadas a objetos para criar um modelo do seu sistema.
- Defina abstrações de alto nível, compartilhe-as e publique-as em sua equipe, empresa ou comunidade.
- Organize suas soluções em módulos lógicos.

- Compartilhe e reutilize sua solução como uma biblioteca.
- Teste seu código de infraestrutura usando protocolos padrão do setor.
- Use seu fluxo de trabalho de revisão de código existente.

O objetivo do AWS Solutions Constructs é reduzir a complexidade e a lógica de cola necessárias ao integrar padrões comuns bem arquitetados para atingir suas metas de solução na AWS.

Conceitos básicos do AWS Solutions Constructs

Este tópico descreve como instalar e configurar o AWS Cloud Development Kit (AWS CDK), o AWS Solutions Constructs e criar seu primeiro aplicativo AWS CDK usando os padrões do AWS Solutions Constructs.

> **Note**
>
> Constritos de soluções da AWS são compatíveis com versões do AWS CDK ≥ 1.46.0.

> **Tip**
>
> Quer cavar mais fundo? Experimente aWorkshop CDKpara uma excursão mais aprofundada de um projeto do mundo real.

> **Tip**
>
> Para obter mais informações sobre como começar a usar o Cloud Development Kit AWS (AWS CDK), consulte aGuia do desenvolvedor do AWS CDK.

Prerequisites

O AWS Solutions Constructs é construído com base no AWS CDK, portanto, você precisa instalar o Node.js (>= 10.3.0), mesmo aqueles que trabalham em idiomas diferentes do TypeScript ou JavaScript. Isso ocorre porque oAWS CDKe AWS Solutions Constructs são desenvolvidos em TypeScript e executados em Node.js. As ligações para outros idiomas suportados usam este backend e conjunto de ferramentas.

Você deve fornecer suas credenciais e uma região da AWS para usar o AWS CDK CLI, conforme descrito em Especificando suas credenciais e região.

Outros pré-requisitos dependem da sua linguagem de desenvolvimento, como se segue.

Linguagem	Pré-requisitos
P	Python >= 3.6
T t	Texto TypeScript >= 2.7
Ja	Java >= 1.8

Instalar o AWS CDK

Para instalar e configurar o AWS CDK, consulte o AWS CDK Developer Guide -Instalar o AWS CDK.

Trabalhar com construções de soluções da AWS

O fluxo de trabalho típico para criar um novo aplicativo ao trabalhar com o AWS Solutions Constructs segue a mesma abordagem que o AWS CDK.

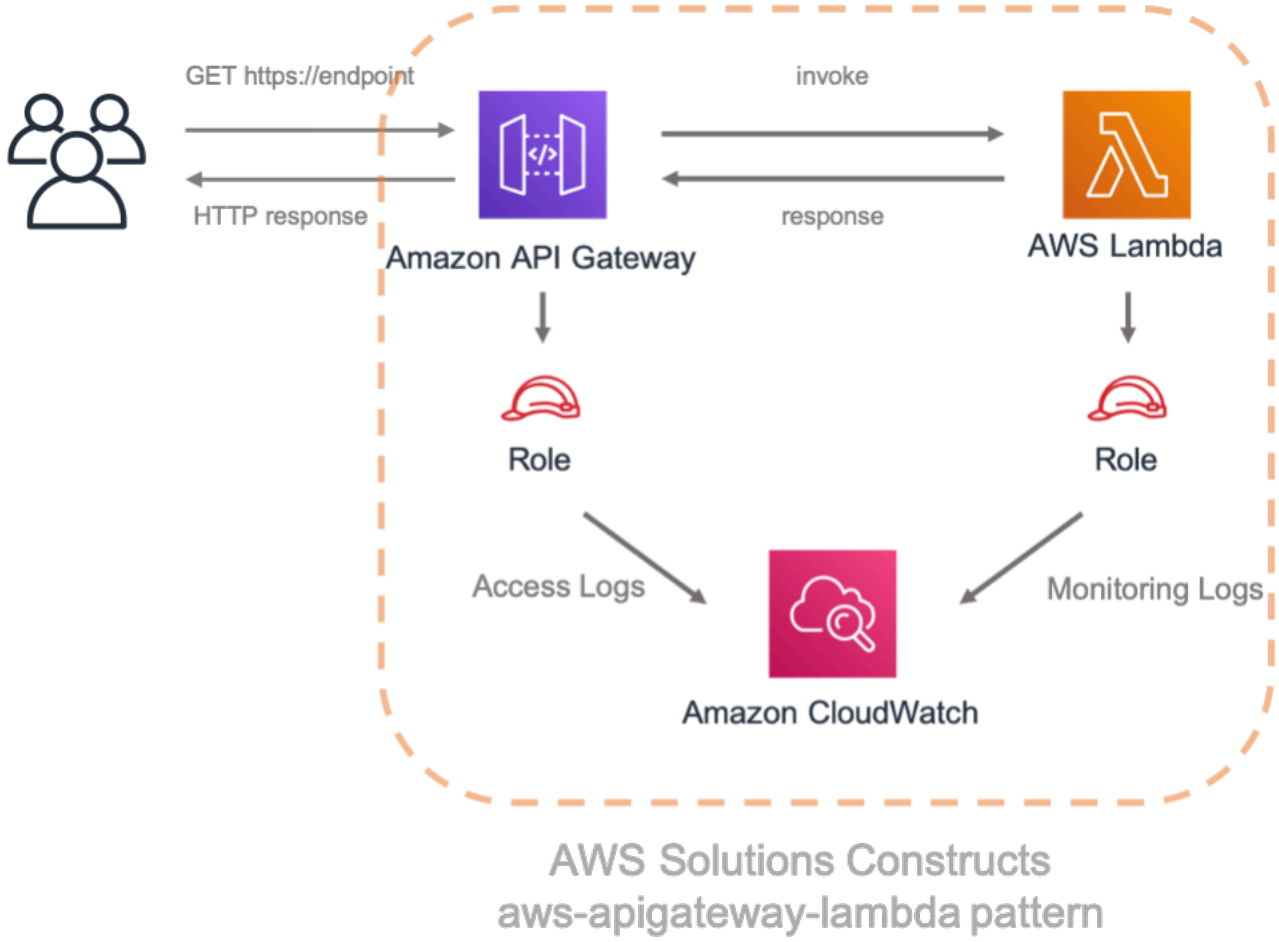
1. Crie o diretório do aplicativo.
2. Inicialize o aplicativo.
3. Adicione as dependências de padrão do AWS Solutions Constructs.
4. Adicione código adicional ao aplicativo.
5. Compilar o aplicativo, se necessário.
6. Implante os recursos definidos no aplicativo.
7. Testar o aplicativo.

Se houver algum problema, faça o loop através de modificar, compilar (se necessário), implantar e testar novamente.

Passo a passo - Parte 1

> **Note**
>
> Os constructos de soluções da AWS são compatíveis com as versões CDK da AWS ≥ 1.46.0.

Este tutorial mostra como criar e implantar um aplicativo simples “Hello Constructs” da AWS CDK que usa um padrão do AWS Solutions Constructs, desde a inicialização do projeto até a implantação do modelo resultante do AWS CloudFormation. O aplicativo Hello Constructs criará a seguinte solução simples:



Hello constructos

Vamos começar a criar nosso primeiro aplicativo AWS CDK usando desenvolvimento baseado em padrões.

> **Note**
>
> Esta é uma modificação de exemplo do`Hello CDK!`do[Workshop CDK](). Se esta é a primeira vez que você usa o AWS CDK, recomendamos começar com este workshop para um passo a passo prático e como aproveitar o CDK na criação de um projeto do mundo real.

Criando o diretório de aplicativos e inicializando o CDK da AWS

Crie um diretório para seu aplicativo CDK e, em seguida, crie um aplicativo AWS CDK nesse diretório.

TypeScript

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

Python

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

> **Tip**
>
> Agora é um bom momento para abrir o projeto no seu IDE favorito e explorar. Para saber mais sobre a estrutura do projeto, selecione o link apropriado:
>
> - TypeScript
> - Python

Atualizar dependências da base do projeto

> **Warning**
>
> Para garantir a funcionalidade adequada, os AWS Solutions Constructs e os pacotes de CDK da AWS devem usar o mesmo número de versão em seu projeto. Por exemplo, se você estiver usando AWS Solutions Constructs v.1.52.0, você também deve usar o AWS CDK v.1.52.0.

> **Tip**
>
> Anote a versão mais recente do AWS Solutions Constructs e aplique esse número de versão ao`VERSION_NUMBER`Respaços reservados nas etapas abaixo (para construções de soluções da AWS e pacotes CDK da AWS). Para verificar todas as versões públicas da biblioteca de Construtos,Clique aqui.

TypeScript

Edite a`package.json`com as seguintes informações:

```
"devDependencies": {
  "@aws-cdk/assert": "VERSION_NUMBER",
  "@types/jest": "^24.0.22",
  "@types/node": "10.17.5",
  "jest": "^24.9.0",
  "ts-jest": "^24.1.0",
  "aws-cdk": "VERSION_NUMBER",
  "ts-node": "^8.1.0",
  "typescript": "~3.7.2"
},
"dependencies": {
  "@aws-cdk/core": "VERSION_NUMBER",
  "source-map-support": "^0.5.16"
}
```

Python

Edite `asetup.py`com as seguintes informações:

```
install_requires=[
    "aws-cdk.core==VERSION_NUMBER",
],
```

Instale as dependências de base de projetos.

TypeScript

```
npm install
```

Python

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

Crie e execute o aplicativo e confirme se ele cria uma pilha vazia.

TypeScript

```
npm run build
cdk synth
```

Python

```
cdk synth
```

Você deve ver uma pilha como a seguinte, em que`CDK-VERSION`é a versão do CDK. (Sua saída pode diferir ligeiramente do que é mostrado aqui.)

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/10.17.0
```

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/3.7.7
```

Código de manipulador do Lambda

Vamos começar com o código de manipulador do AWS Lambda.

Criar um diretório do`lambda`na raiz da árvore de seu projeto.

TypeScript

Adicionar um arquivo chamado`lambda/hello.js`com o seguinte conteúdo:

```
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

Adicionar um arquivo chamado`lambda/hello.py`com o seguinte conteúdo:

```
import json

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

Esta é uma função simples do Lambda que retorna o texto “Olá, Constructs! Você apertou [url path]”. A saída da função também inclui o código de status HTTP e cabeçalhos HTTP. Eles são usados pelo API Gateway para formular a resposta HTTP para o usuário.

Este Lambda é fornecido em JavaScript. Para obter mais informações sobre como escrever funções do Lambda em seu idioma de escolha, consulte oDocumentação do AWS Lambda.

Instale as dependências do AWS CDK e do AWS Solutions Constructs

O AWS Solutions Constructs é fornecido com uma extensa biblioteca de construções. A biblioteca é dividida em módulos, um para cada padrão bem arquitetado. Por exemplo, se você quiser definir uma API do Amazon API Gateway Rest para uma função do AWS Lambda, precisaremos usar o`aws-apigateway-lambda`Biblioteca de padrões.

Também precisamos adicionar a biblioteca de construções do AWS Lambda e do Amazon API Gateway a partir do AWS CDK.

Instale o módulo do AWS Lambda e todas as suas dependências em nosso projeto:

> **Note**
>
> Lembre-se de substituir a versão correta e correspondente a ser usada para construções de soluções da AWS e para o CDK da AWS no`VERSION_NUMBER`campos de espaço reservado para cada comando. Versões incompatíveis entre pacotes podem causar erros.

TypeScript

```
npm install -s @aws-cdk/aws-lambda@VERSION_NUMBER
```

Python

```
pip install aws_cdk.aws_lambda==VERSION_NUMBER
```

Em seguida, instale o módulo Amazon API Gateway e todas as suas dependências em nosso projeto:

TypeScript

```
npm install -s @aws-cdk/aws-apigateway@VERSION_NUMBER
```

Python

```
pip install aws_cdk.aws_apigateway==VERSION_NUMBER
```

Por fim, instale os Construtos de soluções da AWS`aws-apigateway-lambda`e todas as suas dependências em nosso projeto:

TypeScript

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

Python

```
pip install aws_solutions_constructs.aws_apigateway_lambda==VERSION_NUMBER
```

Adicione um padrão do Amazon API Gateway/AWS Lambda à sua pilha

Agora, vamos definir o padrão AWS Solutions Constructs para implementar um Amazon API Gateway com um proxy AWS Lambda.

TypeScript

Editar o arquivo`lib/hello-constructs.ts`com o seguinte:

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

export class HelloConstructsStack extends cdk.Stack {
  constructor(scope: cdk.Construct, id: string, props?: cdk.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_12_X,
        handler: 'hello.handler'
      },
```

```
      apiGatewayProps: {
        defaultMethodOptions: {
          authorizationType: api.AuthorizationType.NONE
        }
      }
    };

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

Python

Editar o arquivo`hello_constructs/hello_constructs_stack.py`com o seguinte:

```
from aws_cdk import (
    aws_lambda as _lambda,
    aws_apigateway as apigw,
    core,
)

from aws_solutions_constructs import (
    aws_apigateway_lambda as apigw_lambda
)

class HelloConstructsStack(core.Stack):

    def __init__(self, scope: core.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.asset('lambda'),
                handler='hello.handler',
            ),
            api_gateway_props=apigw.RestApiProps(
                default_method_options=apigw.MethodOptions(
                    authorization_type=apigw.AuthorizationType.NONE
```

```
                )
            )
        )
```

É isso. Isso é tudo o que você precisa fazer para definir um API Gateway que proxies todas as solicitações para uma função do AWS Lambda. Vamos comparar nossa nova pilha com a original:

TypeScript

```
npm run build
cdk diff
```

Python

```
cdk diff
```

O resultado deve ser semelhante ao seguinte:

```
Stack HelloConstructsStack
IAM Statement Changes
###########################################################################################
#   # Resource                    # Effect # Action                      # Principal
                # Condition                   #
###########################################################################################
# + # ${LambdaFunction.Arn}       # Allow  # lambda:InvokeFunction       #
 Service:apigateway.amazonaw # "ArnLike": {                 #
#   #                             #        #                             # s.com
                #   "AWS:SourceArn": "arn:${AW #
#   #                             #        #                             #
                # S::Partition}:execute-api:${ #
#   #                             #        #                             #
                # AWS::Region}:${AWS::AccountI #
#   #                             #        #                             #
                # d}:${RestApi0C43BF4B}/${Rest #
```

```
#   #                               #        #                               #
                # Api/DeploymentStage.prod}/*/ #
#   #                               #        #                               #
                # {proxy+}"                    #
#   #                               #        #                               #
                # }                            #
# + # ${LambdaFunction.Arn}       # Allow  # lambda:InvokeFunction       #
 Service:apigateway.amazonaw # "ArnLike": {                 #
#   #                               #        #                               # s.com
                #   "AWS:SourceArn": "arn:${AW #
#   #                               #        #                               #
                # S::Partition}:execute-api:${ #
#   #                               #        #                               #
                # AWS::Region}:${AWS::AccountI #
#   #                               #        #                               #
                # d}:${RestApi0C43BF4B}/test-i #
#   #                               #        #                               #
                # nvoke-stage/*/{proxy+}"      #
#   #                               #        #                               #
                # }                            #
# + # ${LambdaFunction.Arn}       # Allow  # lambda:InvokeFunction       #
 Service:apigateway.amazonaw # "ArnLike": {                 #
#   #                               #        #                               # s.com
                #   "AWS:SourceArn": "arn:${AW #
#   #                               #        #                               #
                # S::Partition}:execute-api:${ #
#   #                               #        #                               #
                # AWS::Region}:${AWS::AccountI #
#   #                               #        #                               #
                # d}:${RestApi0C43BF4B}/${Rest #
#   #                               #        #                               #
                # Api/DeploymentStage.prod}/*/ #
#   #                               #        #                               #
                # "                            #
#   #                               #        #                               #
                # }                            #
# + # ${LambdaFunction.Arn}       # Allow  # lambda:InvokeFunction       #
 Service:apigateway.amazonaw # "ArnLike": {                 #
#   #                               #        #                               # s.com
                #   "AWS:SourceArn": "arn:${AW #
#   #                               #        #                               #
                # S::Partition}:execute-api:${ #
#   #                               #        #                               #
                # AWS::Region}:${AWS::AccountI #
```

```
#   #                               #        #                              #
                # d}:${RestApi0C43BF4B}/test-i #
#   #                               #        #                              #
                # nvoke-stage/*/"              #
#   #                               #        #                              #
                # }                            #
##########################################################################################
# + # ${LambdaFunctionServiceRole # Allow  # sts:AssumeRole              #
 Service:lambda.amazonaws.co #                                #
#   # .Arn}                         #        #                              # m
                #                              #
##########################################################################################
# + # ${LambdaRestApiCloudWatchRo # Allow  # sts:AssumeRole              #
 Service:apigateway.amazonaw #                                #
#   # le.Arn}                       #        #                              # s.com
                #                              #
##########################################################################################
# + # arn:aws:logs:${AWS::Region} # Allow  # logs:CreateLogGroup         # AWS:
${LambdaRestApiCloudWat #                                 #
#   # :${AWS::AccountId}:*          #        # logs:CreateLogStream        # chRole}
                #                              #
#   #                               #        # logs:DescribeLogGroups      #
                #                              #
#   #                               #        # logs:DescribeLogStreams     #
                #                              #
#   #                               #        # logs:FilterLogEvents        #
                #                              #
#   #                               #        # logs:GetLogEvents           #
                #                              #
#   #                               #        # logs:PutLogEvents           #
                #                              #
##########################################################################################
# + # arn:aws:logs:${AWS::Region} # Allow  # logs:CreateLogGroup         # AWS:
${LambdaFunctionService #                                 #
#   # :${AWS::AccountId}:log-grou #        # logs:CreateLogStream        # Role}
                #                              #
#   # p:/aws/lambda/*               #        # logs:PutLogEvents           #
                #                              #
##########################################################################################
(NOTE: There may be security-related changes not in this list. See https://github.com/
aws/aws-cdk/issues/1299)

Parameters
```

```
[+] Parameter AssetParameters/
ba91444ebd644d9419e8cfee417f3aaa728507dd428788a2fc40574646c4340a/S3Bucket
 AssetParametersba91444ebd644d9419e8cfee417f3aaa728507dd428788a2fc40574646c4340aS3Bucket9780A3B
 {"Type":"String","Description":"S3 bucket for asset
 \"ba91444ebd644d9419e8cfee417f3aaa728507dd428788a2fc40574646c4340a\""}
[+] Parameter AssetParameters/
ba91444ebd644d9419e8cfee417f3aaa728507dd428788a2fc40574646c4340a/S3VersionKey
 AssetParametersba91444ebd644d9419e8cfee417f3aaa728507dd428788a2fc40574646c4340aS3VersionKey37F
 {"Type":"String","Description":"S3 key for asset version
 \"ba91444ebd644d9419e8cfee417f3aaa728507dd428788a2fc40574646c4340a\""}
[+] Parameter AssetParameters/
ba91444ebd644d9419e8cfee417f3aaa728507dd428788a2fc40574646c4340a/ArtifactHash
 AssetParametersba91444ebd644d9419e8cfee417f3aaa728507dd428788a2fc40574646c4340aArtifactHash801
 {"Type":"String","Description":"Artifact hash for asset
 \"ba91444ebd644d9419e8cfee417f3aaa728507dd428788a2fc40574646c4340a\""}

Conditions
[+] Condition CDKMetadataAvailable: {"Fn::Or":[{"Fn::Or":[{"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::Equals":[{"Ref":"AWS::Region"},"eu-central-1"]}]},{"Fn::Or":[{"Fn::Equals":
[{"Ref":"AWS::Region"},"eu-north-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::Equals":[{"Ref":"AWS::Region"},"us-west-1"]},{"Fn::Equals":
[{"Ref":"AWS::Region"},"us-west-2"]}]}]}

Resources
[+] AWS::Logs::LogGroup ApiGatewayToLambda/ApiAccessLogGroup
 ApiGatewayToLambdaApiAccessLogGroupE2B41502
[+] AWS::IAM::Role LambdaFunctionServiceRole LambdaFunctionServiceRole0C4CDE0B
[+] AWS::Lambda::Function LambdaFunction LambdaFunctionBF21E41F
[+] AWS::ApiGateway::RestApi RestApi RestApi0C43BF4B
[+] AWS::ApiGateway::Deployment RestApi/Deployment
 RestApiDeployment180EC503d2c6df3c8dc8b7193b98c1a0bff4e677
[+] AWS::ApiGateway::Stage RestApi/DeploymentStage.prod
 RestApiDeploymentStageprod3855DE66
[+] AWS::ApiGateway::Resource RestApi/Default/{proxy+} RestApiproxyC95856DD
```

```
[+] AWS::Lambda::Permission RestApi/Default/{proxy+}/ANY/
ApiPermission.HelloConstructsStackRestApiFDB18C2E.ANY..{proxy+}
 RestApiproxyANYApiPermissionHelloConstructsStackRestApiFDB18C2EANYproxyE43D39B3
[+] AWS::Lambda::Permission RestApi/Default/{proxy+}/ANY/
ApiPermission.Test.HelloConstructsStackRestApiFDB18C2E.ANY..{proxy+}
 RestApiproxyANYApiPermissionTestHelloConstructsStackRestApiFDB18C2EANYproxy0B23CDC7
[+] AWS::ApiGateway::Method RestApi/Default/{proxy+}/ANY RestApiproxyANY1786B242
[+] AWS::Lambda::Permission RestApi/Default/ANY/
ApiPermission.HelloConstructsStackRestApiFDB18C2E.ANY..
 RestApiANYApiPermissionHelloConstructsStackRestApiFDB18C2EANY5684C1E6
[+] AWS::Lambda::Permission RestApi/Default/ANY/
ApiPermission.Test.HelloConstructsStackRestApiFDB18C2E.ANY..
 RestApiANYApiPermissionTestHelloConstructsStackRestApiFDB18C2EANY81DBDF56
[+] AWS::ApiGateway::Method RestApi/Default/ANY RestApiANYA7C1DC94
[+] AWS::ApiGateway::UsagePlan RestApi/UsagePlan RestApiUsagePlan6E1C537A
[+] AWS::Logs::LogGroup ApiAccessLogGroup ApiAccessLogGroupCEA70788
[+] AWS::IAM::Role LambdaRestApiCloudWatchRole LambdaRestApiCloudWatchRoleF339D4E6
[+] AWS::ApiGateway::Account LambdaRestApiAccount LambdaRestApiAccount

Outputs
[+] Output RestApi/Endpoint RestApiEndpoint0551178A: {"Value":{"Fn::Join":["",
["https://",{"Ref":"RestApi0C43BF4B"},".execute-api.",{"Ref":"AWS::Region"},".",
{"Ref":"AWS::URLSuffix"},"/",{"Ref":"RestApiDeploymentStageprod3855DE66"},"/"]]}}
```

Isso é legal. Este exemplo simples, com um padrão bem arquitetado do AWS Solutions Constructs, adicionou 21 novos recursos à sua pilha.

Implantação cdk

> **Tip**
>
> Antes de implantar seu primeiro aplicativo CDK da AWS que contém uma função do Lambda, você deve inicializar seu ambiente da AWS. Isso cria um bucket de preparação que o AWS CDK usa para implantar pilhas contendo ativos. Se esta for a primeira vez que você estiver usando o AWS CDK para implantar ativos, será necessário executar `ocdk bootstrap`para implantar a pilha do CDK Toolkit em seu ambiente da AWS.

Pronto para implantar?

```
cdk deploy
```

Saídas da pilha

Quando a implantação estiver concluída, você notará esta linha:

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

Esta é uma saída de pilha que é automaticamente adicionada pelo padrão AWS Solutions Constructs e inclui a URL do endpoint do API Gateway.

Testar seu aplicativo

Vamos tentar acertar esse endpoint com`curl`. Copie o URL e execute (seu prefixo e região provavelmente serão diferentes).

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

O resultado deve ser semelhante ao seguinte:

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

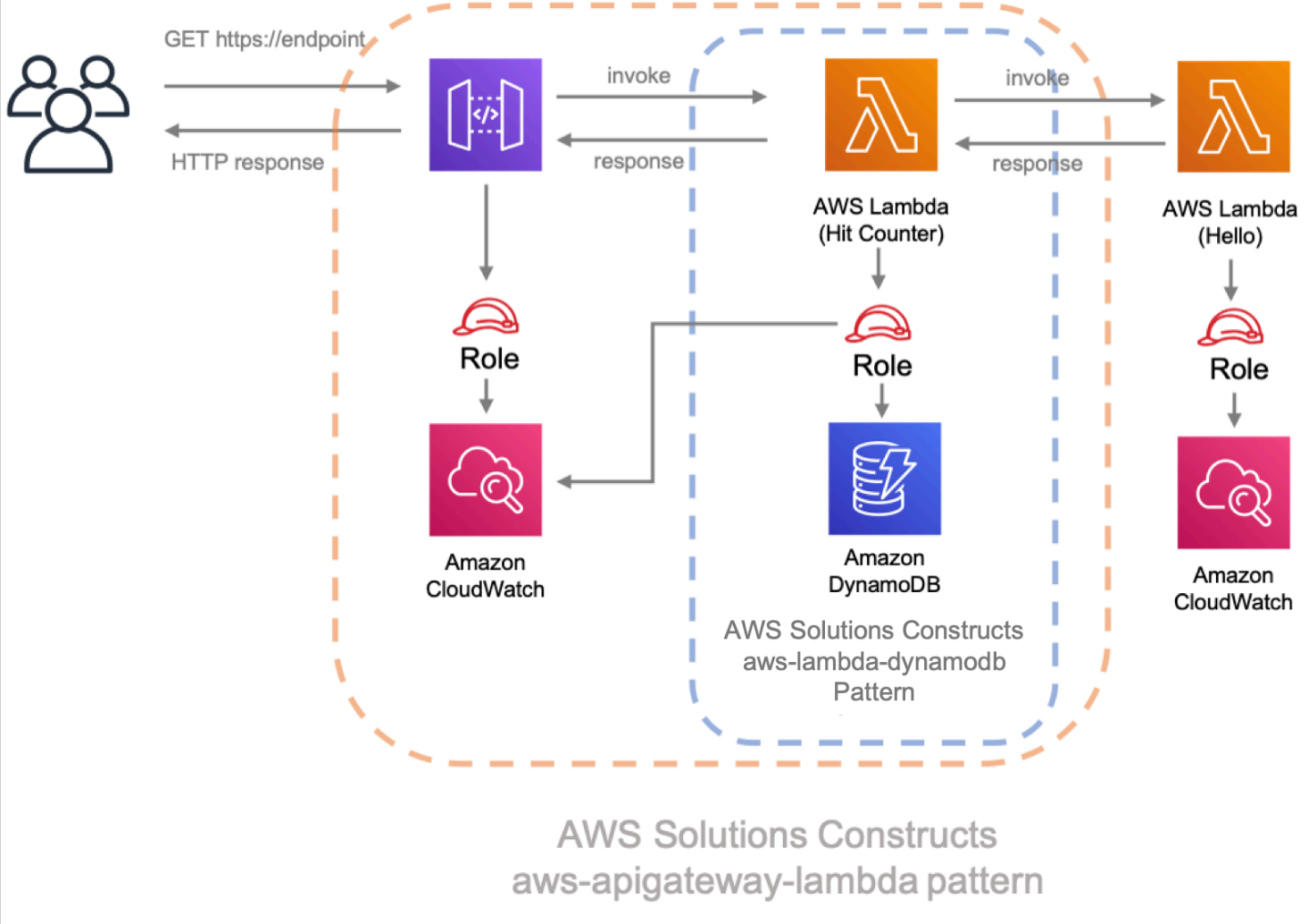
Se esta é a saída que você recebeu, seu aplicativo funciona!

Passo a passo - Parte 2

> **Note**
>
> O AWS Solutions Constructs é compatível com versões ≥ 1.46.0 do AWS.

Este tutorial mostra como modificar o aplicativo “Hello Constructs” criado emparte 1. Nossa modificação adicionará um contador de visitas ao site usando o padrão AWS Lambda ao DynamoDB

de Constructs de soluções da AWS. Modificar o aplicativo Hello Constructs resultará na seguinte solução:



Código do Lambda do contador

Vamos começar escrevendo o código para a função Hit Counter AWS Lambda. Essa função irá:

- incrementar um contador relacionado ao caminho da API em uma tabela do Amazon DynamoDB,
- invocar a função downstream Hello AWS Lambda,
- e retorne a resposta ao usuário final.

TypeScript

Adicionar um arquivo chamado`lambda/hitcounter.js`com o seguinte conteúdo:

```
  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

Adicionar um arquivo chamado`lambda/hitcounter.py`com o seguinte conteúdo:

```
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)
```

Instalar as novas dependências

> **Note**
>
> Lembre-se de substituir a versão correta e correspondente a ser usada para construções de soluções da AWS e para o CDK da AWS no`VERSION_NUMBER`campos de espaço reservado para cada comando. Isso deve ser idêntico ao número de versão usado para dependências na primeira parte deste passo a passo. Versões incompatíveis entre pacotes podem causar erros.

Como de costume, primeiro precisamos instalar as dependências necessárias para a atualização da nossa solução. Primeiro, precisamos instalar a biblioteca de construção do DynamoDB:

TypeScript

```
npm install -s @aws-cdk/aws-dynamodb@VERSION_NUMBER
```

Python

```
pip install aws_cdk.aws_dynamodb==VERSION_NUMBER
```

Por fim, instale os Construtos de soluções da AWS`aws-lambda-dynamodb`e todas as suas dependências em nosso projeto:

TypeScript

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

Python

```
pip install aws_solutions_constructs.aws_lambda_dynamodb==VERSION_NUMBER
```

Defina os recursos

Agora, vamos atualizar nosso código de pilha para acomodar nossa nova arquitetura.

Primeiro, vamos importar nossas novas dependências e mover a função “Olá” para fora do`aws-apigateway-lambda`padrão que criamos na parte 1.

TypeScript

Editar o arquivo`lib/hello-constructs.ts`com o seguinte:

```
import * as cdk from '@aws-cdk/core';
import * as lambda from '@aws-cdk/aws-lambda';
import * as api from '@aws-cdk/aws-apigateway';
import * as dynamodb from '@aws-cdk/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 cdk.Stack {
  constructor(scope: cdk.Construct, id: string, props?: cdk.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);
  }
}
```

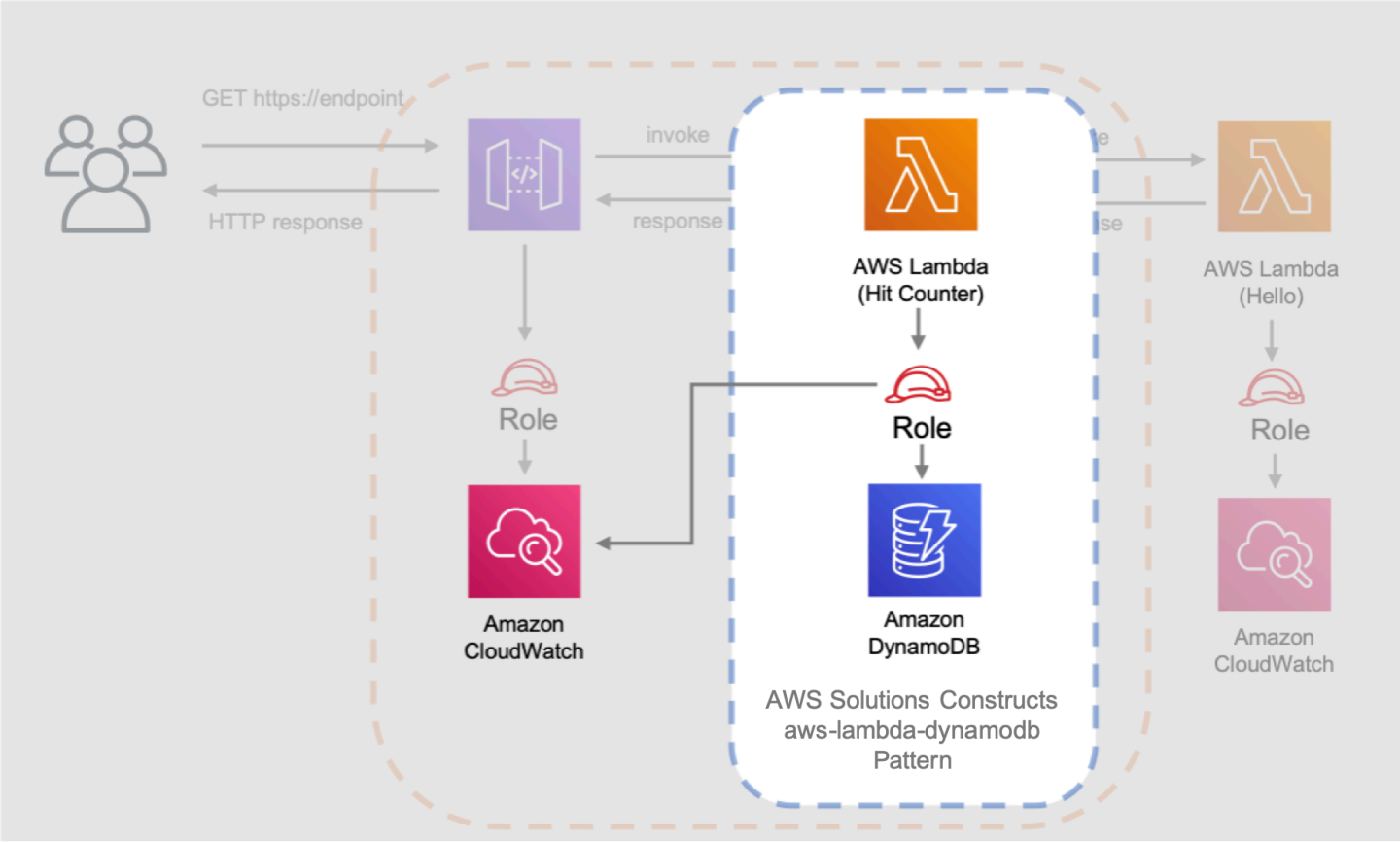
Python

Editar o arquivo`hello_constructs/hello_constructs_stack.py`com o seguinte:

```
from aws_cdk import (
    aws_lambda as _lambda,
    aws_apigateway as apigw,
    aws_dynamodb as ddb,
    core,
)
```

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

class HelloConstructsStack(core.Stack):

    def __init__(self, scope: core.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.asset('lambda'),
        )

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

Em seguida, vamos adicionar o`aws-lambda-dynamodb`padrão para construir o serviço de contador de sucesso para nossa arquitetura atualizada.



AWS Solutions Constructs
aws-apigateway-lambda pattern

A próxima atualização abaixo define as propriedades para o`aws-lambda-dynamodb`definindo a função do AWS Lambda com o manipulador Hit Counter. Além disso, a tabela do Amazon DynamoDB é definida com um nome de`Hits`e uma chave de partição do`path`.

TypeScript

Editar o arquivo`lib/hello-constructs.ts`com o seguinte:

```
import * as cdk from '@aws-cdk/core';
import * as lambda from '@aws-cdk/aws-lambda';
import * as api from '@aws-cdk/aws-apigateway';
import * as dynamodb from '@aws-cdk/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 cdk.Stack {
  constructor(scope: cdk.Construct, id: string, props?: cdk.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.asset(`lambda`),
          runtime: lambda.Runtime.NODEJS_12_X,
          handler: 'hitcounter.handler',
          environment: {
              DOWNSTREAM_FUNCTION_NAME: helloFunc.functionName
          }
      },
      dynamoTableProps: {
          tableName: 'Hits',
          partitionKey: { name: 'path', type: dynamodb.AttributeType.STRING }
      }
    };

    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

Editar o arquivo`hello_constructs/hello_constructs_stack.py`com o seguinte:

```
from aws_cdk import (
    aws_lambda as _lambda,
    aws_apigateway as apigw,
    aws_dynamodb as ddb,
    core,
)

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

class HelloConstructsStack(core.Stack):

    def __init__(self, scope: core.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.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.asset('lambda'),
                handler='hitcounter.handler',
```

```
                environment={
                    'DOWNSTREAM_FUNCTION_NAME': self.hello_func.function_name
                }
            ),
            dynamo_table_props=ddb.TableProps(
                table_name='Hits',
                partition_key={
                    'name': 'path',
                    'type': ddb.AttributeType.STRING
                }
            )
        )

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

Em seguida, precisaremos conceder a função Contador de Hits criada a partir do`aws-lambda-dynamodb`padrão adicionado acima permissão para invocar nossa função Hello.

TypeScript

Editar o arquivo`lib/hello-constructs.ts`com o seguinte:

```
  import * as cdk from '@aws-cdk/core';
import * as lambda from '@aws-cdk/aws-lambda';
import * as api from '@aws-cdk/aws-apigateway';
import * as dynamodb from '@aws-cdk/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 cdk.Stack {
  constructor(scope: cdk.Construct, id: string, props?: cdk.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.asset(`lambda`),
          runtime: lambda.Runtime.NODEJS_12_X,
          handler: 'hitcounter.handler',
          environment: {
              DOWNSTREAM_FUNCTION_NAME: helloFunc.functionName
          }
      },
      dynamoTableProps: {
          tableName: 'Hits',
          partitionKey: { name: 'path', type: dynamodb.AttributeType.STRING }
      }
    };

    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

Editar o arquivo`hello_constructs/hello_constructs_stack.py`com o seguinte:

```
from aws_cdk import (
    aws_lambda as _lambda,
    aws_apigateway as apigw,
    aws_dynamodb as ddb,
    core,
)

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

class HelloConstructsStack(core.Stack):

    def __init__(self, scope: core.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.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.asset('lambda'),
                handler='hitcounter.handler',
                environment={
                    'DOWNSTREAM_FUNCTION_NAME': self.hello_func.function_name
                }
            ),
            dynamo_table_props=ddb.TableProps(
                table_name='Hits',
                partition_key={
                    'name': 'path',
                    'type': ddb.AttributeType.STRING
                }
            )
        )

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

Finalmente, precisamos atualizar nosso original`aws-apigateway-lambda`para utilizar nossa nova função Hit Counter que foi provisionada com o`aws-lambda-dynamodb`Padrão acima.

TypeScript

Editar o arquivo`lib/hello-constructs.ts`com o seguinte:

```
  import * as cdk from '@aws-cdk/core';
import * as lambda from '@aws-cdk/aws-lambda';
import * as api from '@aws-cdk/aws-apigateway';
import * as dynamodb from '@aws-cdk/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 cdk.Stack {
  constructor(scope: cdk.Construct, id: string, props?: cdk.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.asset(`lambda`),
          runtime: lambda.Runtime.NODEJS_12_X,
          handler: 'hitcounter.handler',
          environment: {
              DOWNSTREAM_FUNCTION_NAME: helloFunc.functionName
          }
      },
      dynamoTableProps: {
          tableName: 'Hits',
          partitionKey: { name: 'path', type: dynamodb.AttributeType.STRING }
      }
    };
```

```
    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

Editar o arquivo`hello_constructs/hello_constructs_stack.py`com o seguinte:

```
from aws_cdk import (
    aws_lambda as _lambda,
    aws_apigateway as apigw,
    aws_dynamodb as ddb,
    core,
)

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

class HelloConstructsStack(core.Stack):

    def __init__(self, scope: core.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.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.asset('lambda'),
                handler='hitcounter.handler',
                environment={
                    'DOWNSTREAM_FUNCTION_NAME': self.hello_func.function_name
                }
            ),
            dynamo_table_props=ddb.TableProps(
                table_name='Hits',
                partition_key={
                    'name': 'path',
                    'type': ddb.AttributeType.STRING
                }
            )
        )

        # 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
                )
            )
        )
```

Review as alterações

Vamos construir nosso projeto e revisar as alterações em nossos recursos que acontecerão quando implantarmos isso:

```
npm run build
cdk diff
```

Nossa saída deve ser semelhante a esta:

```
Stack HelloConstructsStack
IAM Statement Changes
#######################################################################################
#   # Resource                          # Effect # Action                          #
 Principal                          # Condition #
#######################################################################################
# + # ${HelloHandler.Arn}               # Allow  # lambda:InvokeFunction            #
 AWS:${LambdaFunctionServiceRole}   #           #
#######################################################################################
# + # ${HelloHandler/ServiceRole.Arn}   # Allow  # sts:AssumeRole                   #
 Service:lambda.amazonaws.com       #           #
#######################################################################################
# + # ${LambdaToDynamoDB/DynamoTable.Ar # Allow  # dynamodb:BatchGetItem            #
 AWS:${LambdaFunctionServiceRole}   #           #
#   # n}                                #        # dynamodb:BatchWriteItem          #
                                    #           #
#   #                                 #        # dynamodb:DeleteItem              #
                                    #           #
#   #                                 #        # dynamodb:GetItem                 #
                                    #           #
#   #                                 #        # dynamodb:GetRecords              #
                                    #           #
#   #                                 #        # dynamodb:GetShardIterator        #
                                    #           #
#   #                                 #        # dynamodb:PutItem                 #
                                    #           #
#   #                                 #        # dynamodb:Query                   #
                                    #           #
#   #                                 #        # dynamodb:Scan                    #
                                    #           #
```

```
#   #                                   #        # dynamodb:UpdateItem              #
                                  #           #
#######################################################################################
IAM Policy Changes
#######################################################################################
#   # Resource                    # Managed Policy ARN
                            #
#######################################################################################
# + # ${HelloHandler/ServiceRole} # arn:${AWS::Partition}:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole #
#######################################################################################
(NOTE: There may be security-related changes not in this list. See https://github.com/
aws/aws-cdk/issues/1299)

Resources
[+] AWS::IAM::Role HelloHandler/ServiceRole HelloHandlerServiceRole11EF7C63
[+] AWS::Lambda::Function HelloHandler HelloHandler2E4FBA4D
[+] AWS::DynamoDB::Table LambdaToDynamoDB/DynamoTable
 LambdaToDynamoDBDynamoTable53C1442D
[+] AWS::IAM::Policy LambdaFunctionServiceRole/DefaultPolicy
 LambdaFunctionServiceRoleDefaultPolicy126C8897
[~] AWS::Lambda::Function LambdaFunction LambdaFunctionBF21E41F
 ## [+] Environment
 #   ## {"Variables":{"DOWNSTREAM_FUNCTION_NAME":
{"Ref":"HelloHandler2E4FBA4D"},"DDB_TABLE_NAME":
{"Ref":"LambdaToDynamoDBDynamoTable53C1442D"}}}
 ## [~] Handler
 #   ## [-] hello.handler
 #   ## [+] hitcounter.handler
 ## [~] DependsOn
     ## @@ -1,3 +1,4 @@
        [ ] [
        [+]   "LambdaFunctionServiceRoleDefaultPolicy126C8897",
        [ ]   "LambdaFunctionServiceRole0C4CDE0B"
        [ ] ]
```

Implantação cdk

Pronto para implantar?

```
cdk deploy
```

Saídas da pilha

Quando a implantação estiver concluída, você notará esta linha:

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

Testar seu aplicativo

Vamos tentar acertar este ponto final com curl. Copie o URL e execute (seu prefixo e região provavelmente serão diferentes).

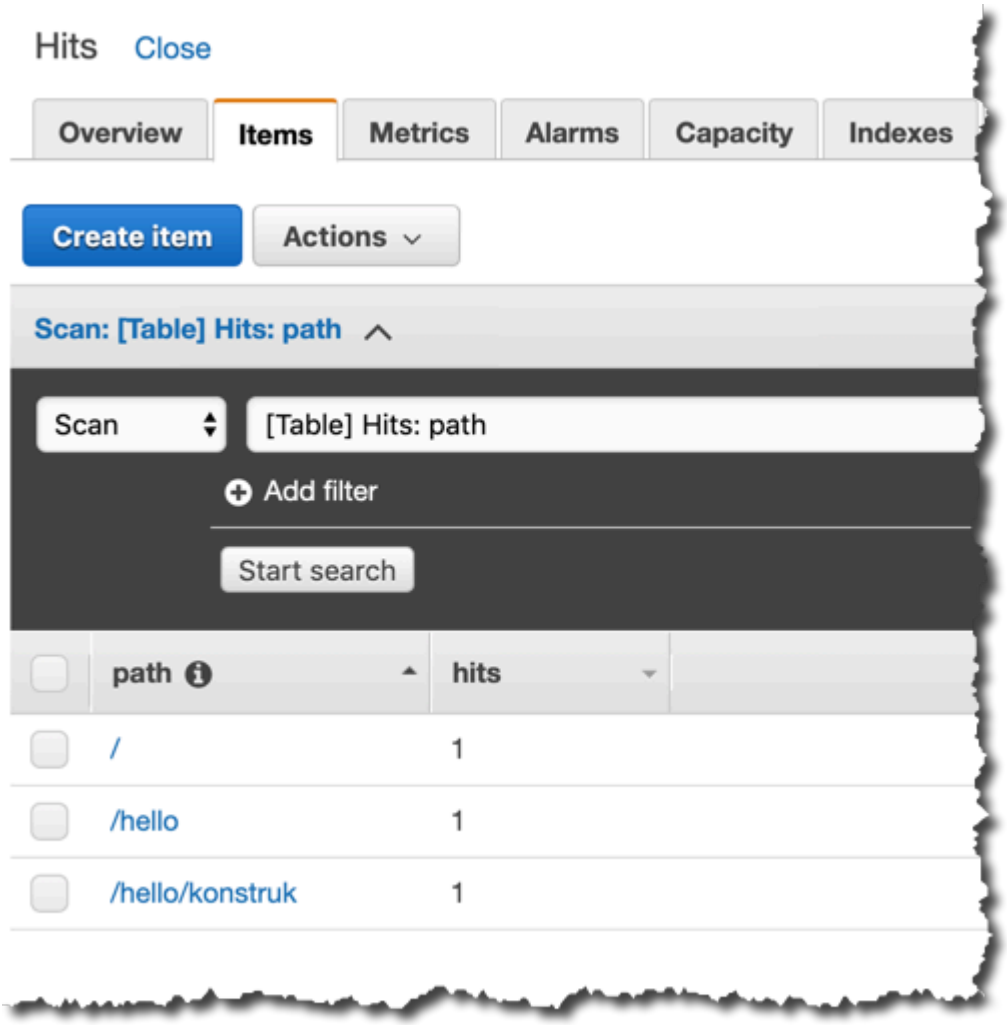
```
curl https://xxxxxxxxxx.execute-api.us-east-1.amazonaws.com/prod/
```

O resultado deve ser semelhante ao seguinte:

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

Agora, vamos revisar o`HitsTabela do Amazon DynamoDB.

1. Acesse o console do DynamoDB.
2. Verifique se você está na Região onde criou a tabela.
3. SelectTabelasno painel de navegação e selecione a caixa de seleçãoHitsTabela INTO.
4. Abra a tabela e selecione “Itens”.
5. Você deve ver quantos hits você tem para cada caminho.



6. Tente acertar um novo caminho e atualize a exibição Itens. Você verá um novo item com um`hits`contagem de um.

Se esta é a saída que você recebeu, seu aplicativo funciona!

Casos de uso do

Esta biblioteca inclui uma coleção de implementações de caso de uso funcional para demonstrar o uso de padrões arquitetônicos de construções. Estes podem ser usados da mesma maneira que os padrões arquitetônicos, e podem ser conceituados como uma abstração adicional de “nível superior” desses padrões. Os seguintes casos de uso são fornecidos como exemplos funcionais:

Site do AWS Static S3

Este padrão de caso de uso (`aws-s3-static-website`) implementa uma distribuição do Amazon CloudFront, bucket do Amazon S3 e recurso personalizado baseado no AWS Lambda para copiar o conteúdo estático do site de demonstração do Wild Rydes (parte do`aws-serverless-web-app`implementação do.

> Código Fonte (aws-s3-static-website)
>
> https://github.com/awslabs/aws-solutions-constructs/tree/master/source/use_cases/aws-s3-static-website

Manipulador de imagens simples sem servidor da AWS

Este padrão de caso de uso (`aws-serverless-image-handler`) implementa uma distribuição do Amazon CloudFront, uma API REST do Amazon API Gateway, uma função do AWS Lambda e permissões/lógicas necessárias para provisionar uma API de manipulador de imagem funcional para servir conteúdo de imagem de um ou mais buckets do Amazon S3 na conta de implantação.

> Código fonte (aws-serverless-image-handler)
>
> https://github.com/awslabs/aws-solutions-constructs/tree/master/source/use_cases/aws-serverless-image-handler

Aplicativo web sem servidor da AWS

Este padrão de caso de uso (`aws-serverless-web-app`) implementa um simples aplicativo web sem servidor que permite aos usuários solicitar passeios de unicórnio da frota Wild Rydes. O aplicativo apresentará aos usuários uma interface de usuário baseada em HTML para indicar o local onde eles gostariam de ser retirados e fará interface no backend com um serviço Web RESTful para enviar a solicitação e enviar um unicórnio próximo. O aplicativo também fornecerá recursos para que os usuários se registrem no serviço e façam login antes de solicitar viagens.

Código fonte (aws-serverless-web-app)

https://github.com/awslabs/aws-solutions-constructs/tree/master/source/use_cases/aws-serverless-web-app

Referência de API

O AWS Solutions Constructs (Constructs) é uma extensão de código aberto do AWS Cloud Development Kit (AWS CDK) que fornece padrões multi-serviço e bem arquitetados para definir rapidamente soluções em código para criar infraestrutura previsível e repetível. O objetivo do Constructs é acelerar a experiência para os desenvolvedores criarem soluções de qualquer tamanho usando definições baseadas em padrões para sua arquitetura.

Os padrões definidos em Constructs são abstrações de alto nível e vários serviços de construções CDK da AWS que têm configurações padrão baseadas em práticas recomendadas bem arquitetadas. A biblioteca é organizada em módulos lógicos usando técnicas orientadas a objetos para criar cada modelo de padrão arquitetônico.

O CDK está disponível nas linguagens a seguir:

- JavaScript, TypeScript (Node.js ≥ 10.3.0)
- Python (Python ≥ 3.6)
- Java (Java ≥ 1,8)

Modules

O AWS Solutions Constructs é organizado em vários módulos. Eles são nomeados assim:

- aws-xxx: Pacote padrão bem arquitetado para os serviços indicados. Este pacote conterá construções que contêm vários módulos de serviço CDK da AWS para configurar o padrão fornecido.
- xxx: Pacotes que não iniciam”aw—“são módulos principais de construção que são usados para configurar padrões de práticas recomendadas para serviços usados dentro da biblioteca de padrões.

Conteúdo do Módulo

Os módulos contêm os seguintes tipos:

- Padrões- Todas as construções multi-serviços de nível superior nesta biblioteca.
- Outros tipos- Todas as classes não construídas, interfaces, estruturas e enums que existem para suportar os padrões.

Os padrões levam um conjunto de propriedades (input) em seu construtor; o conjunto de propriedades (e quais são necessárias) pode ser visto na página de documentação de um padrão.

A página de documentação do padrão também lista os métodos disponíveis para chamar e as propriedades que podem ser usadas para recuperar informações sobre o padrão depois que ele foi instanciado.

aws-apigateway-dynamodb

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_apigateway_dynamodb`
TypeScript	`@aws-solutions-constructs/aws-apigateway-dynamodb`
Java	`software.amazon.awsconstructs.services.apigatewaydynamodb`

Overview

Este AWS Solutions Construct implementa uma API REST do Amazon API Gateway conectada a uma tabela do Amazon DynamoDB.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { ApiGatewayToDynamoDBProps, ApiGatewayToDynamoDB } from "@aws-solutions-
constructs/aws-apigateway-dynamodb";

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

Initializer

```
new ApiGatewayToDynamoDB(scope: Construct, id: string, props:
 ApiGatewayToDynamoDBProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`ApiGatewayToDynamoDBProps`

Adereços de construção de padrão

Nome	Tipo	Descrição
DynamoTableProps	`dynamodb.TableProps`	Os adereços fornecidos pelo usuário opcionais para substituir os adereços padrão da Tabela do DynamoDB
ApigatewayProps?	`api.RestApiProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o API Gateway.
AllowCreateOperation	`boolean`	Se a API Gateway Method for Create operation na tabela do DynamoDB deve ser implantada.

Nome	Tipo	Descrição
CreateRequestTemplate	`string`	Modelo de Solicitação de API Gateway para o método Criar, necessário se AllowCreateOperation definido como true
allowReadOperation	`boolean`	Se a operação API Gateway Method for Read deve ser implantada na tabela do DynamoDB.
AllowUpdateOperation	`boolean`	Se a operação API Gateway Method for Update deve ser implantada na tabela do DynamoDB.
UpdateRequestTemplate	`string`	Modelo de Solicitação do API Gateway para o método Update, necessário se AllowUpdateOperation definido como true
AllowDeleteOperation	`boolean`	Se deve implantar a operação API Gateway Method for Delete na tabela do DynamoDB.
LoggroupProps?	`logs.LogGroupProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o grupo de logs do CloudWatch Logs.

Propriedades de padrão

Nome	Tipo	Descrição
APiGateway	`api.RestApi`	Retorna uma instância da API REST Gateway criada pelo padrão.
ApigatewayCloudWatchRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão que permite o log de acesso da API REST do API Gateway para o CloudWatch.
ApigatewayLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para o qual os logs de acesso à API REST do API Gateway são enviados.
ApigateWayRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão para a API REST do API Gateway.
DynamoTable	`dynamodb.Table`	Retorna uma instância da tabela do DynamoDB criada pelo padrão.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon API Gateway

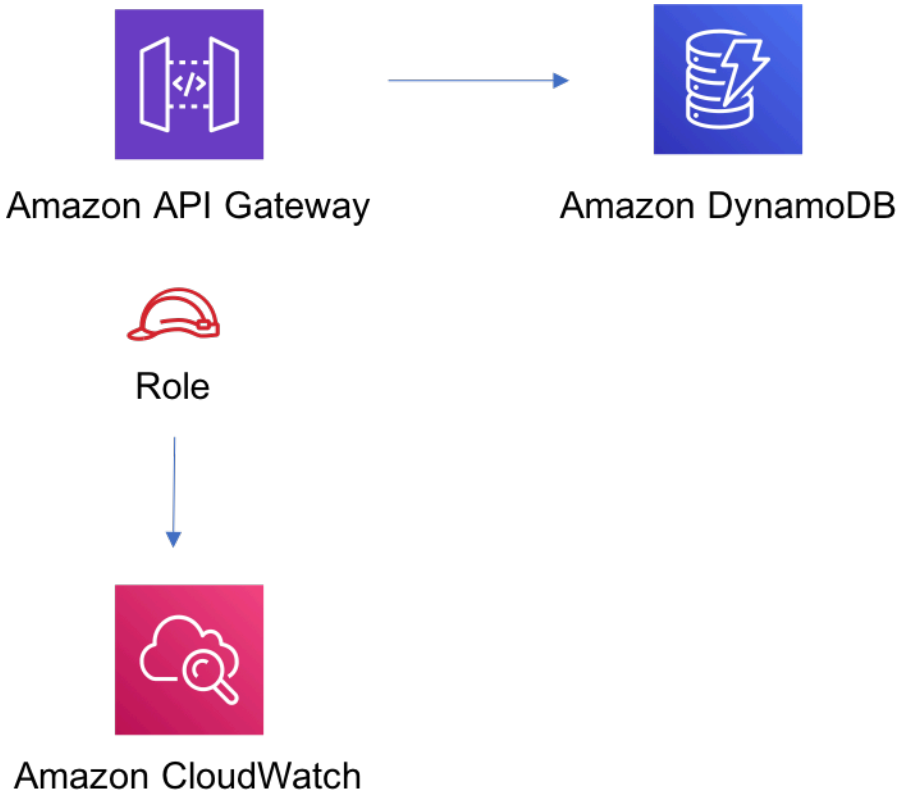
- Implantar um endpoint de API otimizado para bordas
- Ativar o CloudWatch para o API Gateway
- Configurar a função IAM de acesso de menor privilégio para API Gateway

- Defina o AuthorizationType padrão para todos os métodos de API como IAM
- Ativar rastreamento do X-Ray

Tabela do Amazon DynamoDB

- Definir o modo de faturamento da Tabela do DynamoDB como On-Demand (Pagamento por solicitação)
- Habilitar criptografia no lado do servidor para a tabela do DynamoDB usando a chave KMS gerenciada pela AWS
- Cria uma chave de partição chamada 'id' para a tabela do DynamoDB
- Manter a tabela ao excluir a pilha do CloudFormation
- Permita backups contínuos e recuperação point-in-time

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-apigateway-dynamodb

aws-apigateway-muito

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_apigateway_iot`
TypeScript	`@aws-solutions-constructs/aws-apigateway-iot`
Java	`software.amazon.awsconstructs.services.apigatewayiot`

Overview

Este AWS Solutions Construct implementa uma API REST do Amazon API Gateway conectada ao padrão do AWS IoT.

Essa construção cria um proxy HTTPS escalável entre o API Gateway e o AWS IoT. Isso é útil ao querer permitir que dispositivos legados que não suportam o protocolo MQTT ou MQTT/WebSocket interajam com a plataforma AWS IoT.

Essa implementação permite que mensagens somente de gravação sejam publicadas em determinados tópicos MQTT e também suporta atualizações de sombra de dispositivos HTTPS para itens permitidos no registro do dispositivo. Ele não envolve funções do Lambda para mensagens de proxy e, em vez disso, depende da integração direta do API Gateway para o AWS IoT, que suporta mensagens JSON e mensagens binárias.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { ApiGatewayToIot } from '@aws-solutions-constructs/aws-apigateway-iot';

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

Initializer

```
new ApiGatewayToIot(scope: Construct, id: string, props: ApiGatewayToIotProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`ApiGatewayToIotProps`

Props de construção de padrão

Nome	Tipo	Descrição
IoTendpoint	`string`	O subdomínio de endpoint do AWS IoT para integrar o API

Nome	Tipo	Descrição
		Gateway com (por exemplo, a1234567890123-ats).
ApigatewayCreateApikey?	`boolean`	Se definido como`true`, uma chave de API é criada e associada a um UsagePlan . O usuário deve especificar o cabeçalho `x-api-key` ao acessar RestApi. Valor padrão definido como`false`.
ApigatewayExecutionRole?	`iam.Role`	A função do IAM usada pelo API Gateway para acessar o AWS IoT. Se não for especific ado, uma função padrão será criada com acesso curinga ('*') a todos os tópicos e coisas.
ApigatewayProps?	`api.restApiProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para a API REST do API Gateway.
LoggroupProps?	`logs.LogGroupProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o grupo de logs do CloudWatch Logs.

Propriedades de padrão

Nome	Tipo	Descrição
ApiGateway	`api.RestApi`	Retorna uma instância da API REST Gateway criada pelo padrão.

Nome	Tipo	Descrição
ApigatewayCloudWatchRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão que permite o log de acesso da API REST do API Gateway para o CloudWatch.
ApigatewayLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para o qual os logs de acesso à API REST do API Gateway são enviados.
ApigateWayRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão para a API REST do API Gateway.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon API Gateway

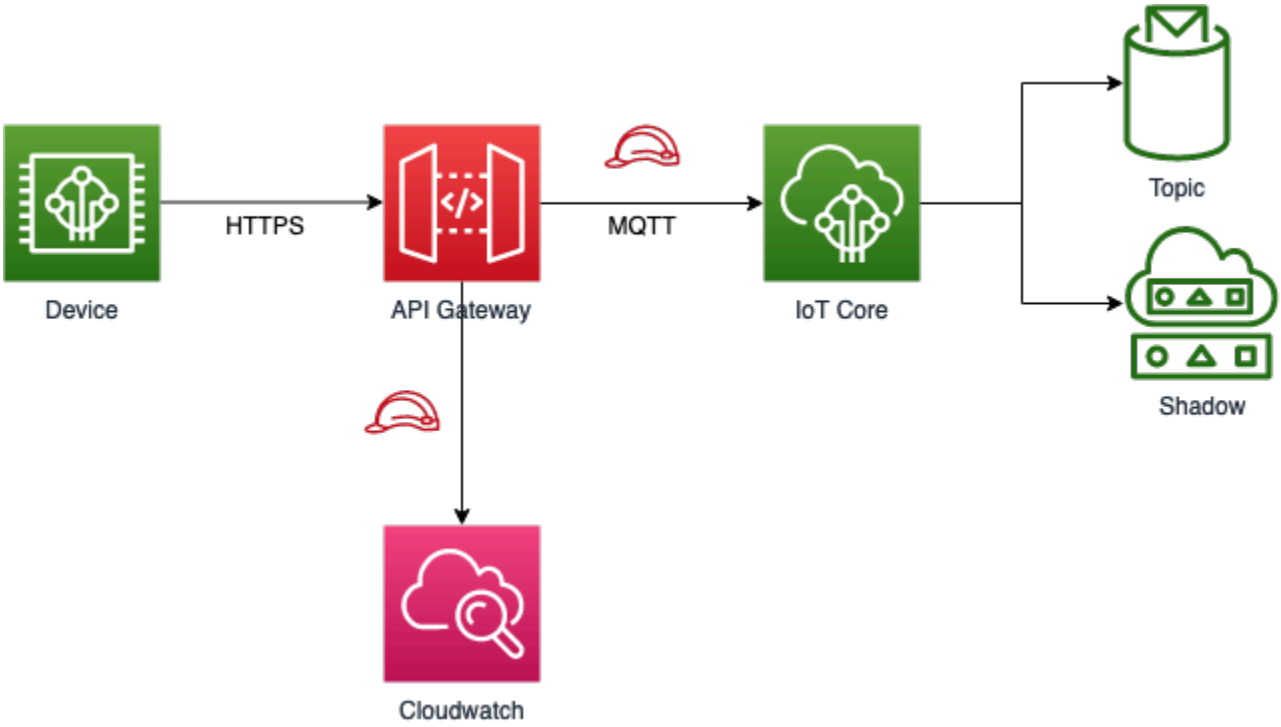
- Implantar um endpoint de API otimizado para bordas
- Cria recursos de API com`POST`Método para publicar mensagens para tópicos do IoT
- Cria recursos de API com`POST`Método para publicar mensagens`ThingShadow`e`NamedShadows`
- Habilitar o log do CloudWatch para o
- Configurar a função do IAM para o API Gateway com acesso a todos os tópicos e coisas
- Defina o AuthorizationType padrão para todos os métodos de API como IAM
- Habilitar X-Ray Rastreamento
- Cria um UsagePlan e associa a`prod`stage

Abaixo está uma descrição dos diferentes recursos e métodos expostos pelo API Gateway após a implantação do Construct. Consulte oExemplospara obter mais informações sobre como testar facilmente esses endpoints usando`curl`.

Método	Recurso	Parâmetro (s) de Consulta	Código (es) de retorno	Descrição
POST	/message/<topics>	qos	200/403/500	Ao chamar este ponto final, você precisa passar os tópicos sobre os quais você gostaria de publicar (por exemplo, `/message/device/foo `).
POST	/shadow/<thingName>	Nenhum	200/403/500	Esta rota permite atualizar o documento sombra de uma coisa, dada a suathingName Usando o tipo de shadow sem nome (clássico). O corpo deve estar em conformidade com a estrutura de sombra normalizada da que inclui umstatenó e associado desiredereportedNó. Consulte oAtualizar

Método	Recurso	Parâmetro (s) de Consulta	Código (es) de retorno	Descrição
				sombras do dispositivoSeção para ver um exemplo.
POST	/shadow/<thingName>/<shadowName>	Nenhum	200/403/500	Esta rota permite atualizar o documento de sombra nomeado de uma coisa, dada a suathingName O e ashadowName usando o tipo de sombra Nomeado. O corpo deve estar em conformidade com a estrutura de sombra normalizada que inclui umstatenó e associado desiredereportedNó. Consulte oAtualizar sombras nomeadasSeção para ver um exemplo.

Architecture



Examples

Os exemplos a seguir funcionam somente com o`API_KEY`, uma vez que a autorização do IAM exige que um token Sigv4 seja especificado também, certifique-se de que o`apiGatewayCreateApiKey`de seus adereços do Construct está definida como`true`ao implantar a pilha, caso contrário, os exemplos abaixo não funcionarão.

Publicar uma mensagem

Você pode usar o`curl`para publicar uma mensagem em diferentes tópicos MQTT usando a API HTTPS. O exemplo abaixo irá postar uma mensagem no`device/foo`Tópico.

```
curl -XPOST https://<stage-id>.execute-api.<region>.amazonaws.com/prod/message/device/
foo -H "x-api-key: <api-key>" -H "Content-Type: application/json" -d '{"Hello":
 "World"}'
```

Observações: Substituir o`stage-id`,`region`, e`api-key`com seus valores de implantação.

Você pode encadear nomes de tópicos no URL e a API aceita até 7 subtópicos nos quais você pode publicar. Por exemplo, o exemplo abaixo publica uma mensagem no tópico`device/foo/bar/abc/xyz`.

```
curl -XPOST https://<stage-id>.execute-api.<region>.amazonaws.com/prod/message/device/
foo/bar/abc/xyz -H "x-api-key: <api-key>" -H "Content-Type: application/json" -d
 '{"Hello": "World"}'
```

Atualizar sombras do dispositivo

Para atualizar o documento sombra associado a uma determinada coisa, você pode emitir uma solicitação de estado de sombra usando um nome de coisa. Consulte o exemplo a seguir sobre como atualizar uma thing shadow.

```
curl -XPOST https://<stage-id>.execute-api.<region>.amazonaws.com/prod/shadow/device1 -
H "x-api-key: <api-key>" -H "Content-Type: application/json" -d '{"state": {"desired":
 { "Hello": "World" }}}'
```

Atualizar sombras nomeadas

Para atualizar o documento sombra associado à sombra nomeada de uma determinada coisa, você pode emitir uma solicitação de estado de sombra usando um nome de coisa e nome de sombra. Consulte o exemplo a seguir sobre como atualizar uma sombra nomeada.

```
curl -XPOST https://<stage-id>.execute-api.<region>.amazonaws.com/prod/shadow/device1/
shadow1 -H "x-api-key: <api-key>" -H "Content-Type: application/json" -d '{"state":
 {"desired": { "Hello": "World" }}}'
```

Enviando cargas binárias

É possível enviar uma carga binária para a API proxy, até o serviço AWS IoT. No exemplo a seguir, enviamos o conteúdo do`README.md`associado a este módulo (tratado como dados binários) para`device/foo`Usando o`application/octet-stream`Tipo de conteúdo.

```
curl -XPOST https://<stage-id>.execute-api.<region>.amazonaws.com/prod/message/device/
foo/bar/baz/qux -H "x-api-key: <api-key>" -H "Content-Type: application/octet-stream"
 --data-binary @README.md
```

Observações: Execute este comando enquanto estiver no diretório deste projeto. Em seguida, você pode testar o envio de outros tipos de arquivos binários do seu sistema de arquivos.

GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-apigateway-iot

aws-apigateway-kinesisstreams

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àSemantic versionModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_apigateway_kinesisstreams`
TypeScript	`@aws-solutions-constructs/aws-apigateway-kinesisstreams`

Linguagem	Pacote
Java	`software.amazon.awsconstructs.services.apigatewaykinesisstreams`

Overview

Esse padrão implementa uma API REST do Amazon API Gateway conectada a um stream de dados do Amazon Kinesis.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { ApiGatewayToKinesisStreams, ApiGatewayToKinesisStreamsProps } from '@aws-solutions-constructs/aws-apigateway-kinesisstreams';

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

Initializer

```
new ApiGatewayToKinesisStreams(scope: Construct, id: string, props:
 ApiGatewayToKinesisStreamsProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`ApiGatewayToKinesisStreamsProps`

Adereços de construção de padrão

Nome	Tipo	Descrição
ApigatewayProps?	`api.RestApiProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para a API REST do API Gateway.
PutRecordRequestTemplate?	`string`	Modelo de solicitação do API Gateway para a ação PutRecord. Se não for fornecido, um padrão será usado.
PutRecordRequestModel?	`api.ModelOptions`	Modelo de solicitação do API Gateway para a ação PutRecord. Se não for fornecido, um padrão será criado.
PutRecordsRequestTemplate?	`string`	Modelo de solicitação do API Gateway para a ação PutRecords. Se não for fornecido, um padrão será usado.
PutRecordRequestModel?	`api.ModelOptions`	Modelo de solicitação do API Gateway para a ação PutRecords. Se não for fornecido, um padrão será criado.
ExistingStreamobj?	`kinesis.Stream`	Instância existente do Kinesis Stream, fornecendo tanto isso quanto`kinesisSt`

Nome	Tipo	Descrição
		`reamProps` causará um erro.
KinesisStreamprops?	`kinesis.StreamProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o stream do Kinesis.
LoggroupProps?	`logs.LogGroupProps`	Props fornecidos pelo usuário para substituir os adereços padrão para o grupo de logs do CloudWatch Logs.

Propriedades do padrão

Nome	Tipo	Descrição
APigGateway	`api.RestApi`	Retorna uma instância da API REST Gateway criada pelo padrão.
ApigateWayRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão para a API REST do API Gateway.
ApigatewayCloudWatchRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão que permite o log de acesso da API REST do API Gateway para o CloudWatch.
ApigatewayLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para o qual os logs de

Nome	Tipo	Descrição
		acesso à API REST do API Gateway são enviados.
KinesisStream	`kinesis.Stream`	Retorna uma instância do stream do Kinesis criado pelo padrão.

Uso da API de amostra

Método	O caminho da solicitação	Corpo da solicitação	Ação da Fila	Descrição
POST	`/record`	`{ "data": "Hello World!", "partitionKey": "pk001" }`	`kinesis:PutRecord`	Grava um único registro de dados no stream.
POST	`/records`	`{ "records": [ { "data": "abc", "partitionKey": "pk001" }, { "data": "xyz",`	`kinesis:PutRecords`	Grava vários registros de dados no fluxo em uma única chamada.

Método	O caminho da solicitação	Corpo da solicitação	Ação da Fila	Descrição
		`"partitionKey": "pk001" } ] }`		

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

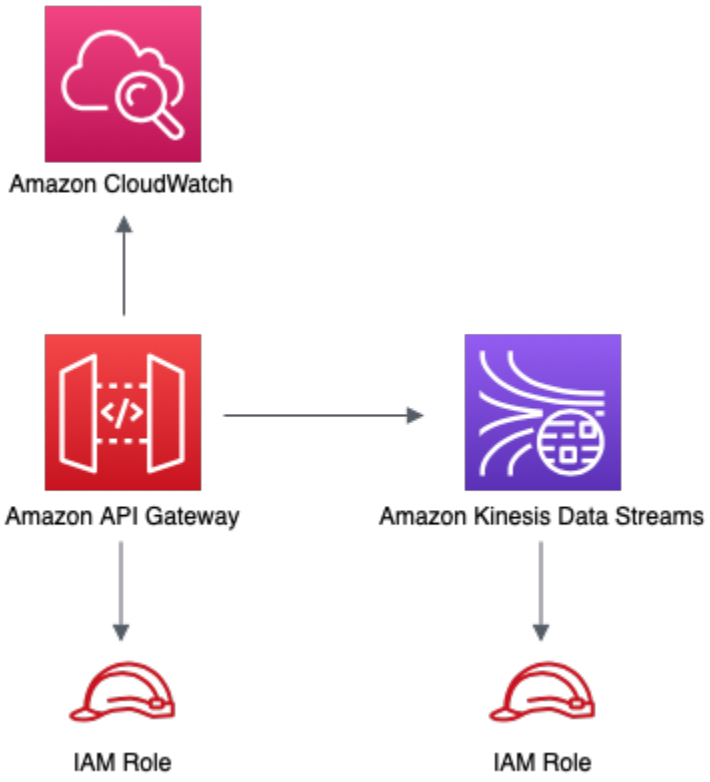
Amazon API Gateway

- Implantar um endpoint de API otimizado para bordas.
- Ative o log do CloudWatch para API Gateway
- Configurar a função do IAM de acesso de menor privilégio para API Gateway.
- Defina o AuthorizationType padrão para todos os métodos de API como IAM.
- Ativar rastreamento do X-Ray.
- Validar o corpo da solicitação antes de passar dados para o Kinesis.

Amazon Kinesis Data Stream

- Configurar a função do IAM de acesso de menor privilégio para stream do Kinesis.
- Ative a criptografia do lado do servidor para o Kinesis Stream usando a chave KMS gerenciada da AWS.

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-apigateway-kinesisstreams

aws-apigateway-lambda

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_apigateway_lambda`
TypeScript	`@aws-solutions-constructs/aws-apigateway-lambda`
Java	`software.amazon.awsconstructs.services.apigatewaylambda`

Overview

Este AWS Solutions Construct implementa uma API REST do Amazon API Gateway conectada a uma função do AWS Lambda.

Aqui está uma definição de padrão implantável mínima no TypeScript:

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

new ApiGatewayToLambda(this, 'ApiGatewayToLambdaPattern', {
    lambdaFunctionProps: {
      runtime: lambda.Runtime.NODEJS_14_X,
      // This assumes a handler function in lib/lambda/index.js
      code: lambda.Code.fromAsset(`${__dirname}/lambda`),
      handler: 'index.handler'
    }
});
```

Initializer

```
new ApiGatewayToLambda(scope: Construct, id: string, props: ApiGatewayToLambdaProps);
```

Parâmetros

- escopoConstruct
- idstring
- propsApiGatewayToLambdaProps

Adereços de construção

Nome	Tipo	Descrição
ExistingAmbdaobj?	lambda.Function	Instância existente do objeto Lambda Function, fornecendo o tanto isso elambdaFunctionProps causará um erro.
LambdaFunctionProps?	lambda.FunctionProps	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se umexistingLambdaObj é fornecido.
ApigatewayProps?	api.LambdaRestApiProps	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para a API.
LoggroupProps?	logs.LogGroupProps	Aderos opcionais fornecidos pelo usuário para substituir os adereços padrão para o grupo de logs do CloudWatch Logs.

Propriedades de padrão

Nome	Tipo	Descrição
ApigatewayCloudWatchRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão que permite o log de acesso da API REST do API Gateway para o CloudWatch.
ApigatewayLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para o qual os logs de acesso à API REST do API Gateway são enviados.
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.
ApiGateway	`api.LambdaRestApi`	Retorna uma instância da API REST Gateway criada pelo padrão.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

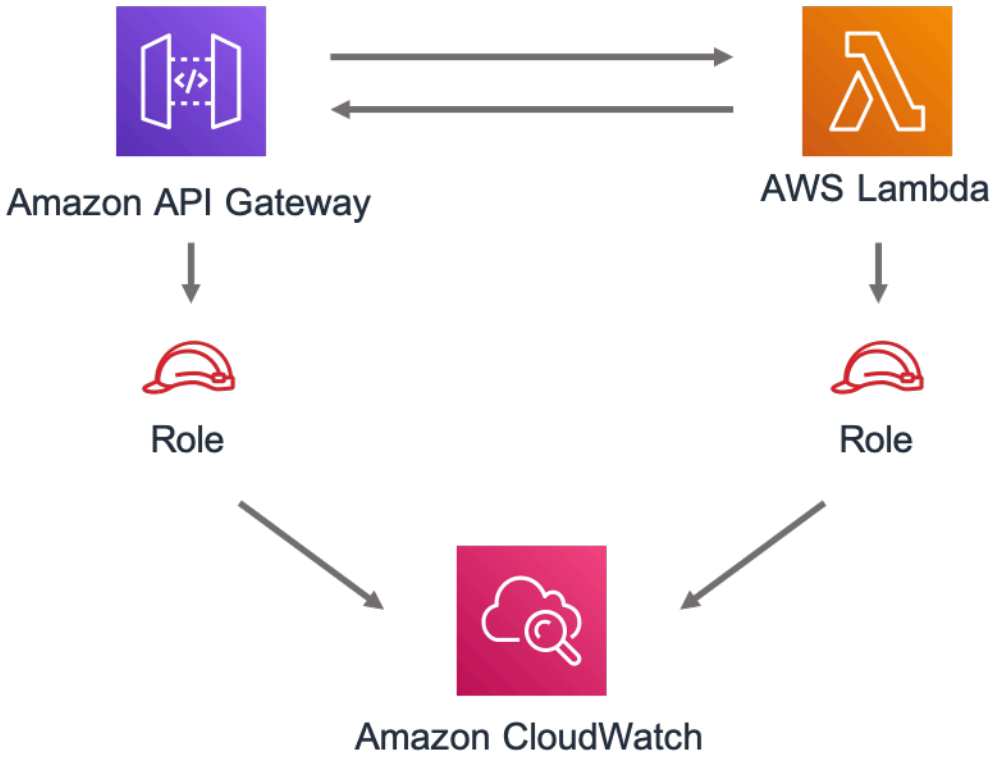
Amazon API Gateway

- Implantar um endpoint de API otimizado para bordas
- Habilitar o log do CloudWatch para o
- Configurar a função IAM de acesso de menor privilégio para API Gateway
- Defina o AuthorizationType padrão para todos os métodos de API como IAM
- Ativar rastreamento do X-Ray
- Definição de variáveis de ambiente:
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Ativar a reutilização de conexões com a função Keep-Alive para NodeJS Lambda
- Ativar rastreamento do X-Ray

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

	@aws -solutions-constructs/aws-apigateway-lambda

aws-apigateway-sagemakerendpoint

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_apigateway_sagemakerendpoint`
TypeScript	`@aws-solutions-constructs/aws-apigateway-sagemakerendpoint`
Java	`software.amazon.awsconstructs.services.apigatewaysagemakerendpoint`

Overview

Este AWS Solutions Construct implementa uma API REST do Amazon API Gateway conectada a um endpoint do Amazon SageMaker.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
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 $items.split(",") )
    {"in0": [$user_id], "in1": [$item]}#if( $foreach.hasNext ),#end
    $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
});
```

Initializer

```
new ApiGatewayToSageMakerEndpoint(scope: Construct, id: string, props:
 ApiGatewayToSageMakerEndpointProps);
```

Parâmetros

- escopoConstruct
- idstring
- propsApiGatewayToSageMakerEndpointProps

Adereços de construção de padrão

Nome	Tipo	Descrição
ApigatewayProps?	api.RestApiProps	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para a API REST do API Gateway.
ApigatewayExecutionRole?	iam.Role	Função do IAM usada pelo API Gateway para chamar

Nome	Tipo	Descrição
		o endpoint do SageMaker. Se não for especificado, uma função padrão será criada com acesso a `endpointName`.
EndpointName	`string`	Nome do ponto de extremidade de inferência do SageMaker implantado.
ResourceName?	`string`	Nome do recurso opcional onde o método GET estará disponível.
resourcePath	`string`	Caminho do recurso para o método GET. A variável definida aqui pode ser referenciada em `requestMappingTemplate`.
RequestMappingTemplate	`string`	Modelo de mapeamento para converter solicitações GET recebidas na API REST em solicitações POST esperadas pelo endpoint SageMaker.
ResponseMappingTemplate?	`string`	Modelo de mapeamento opcional para converter as respostas recebidas do endpoint do SageMaker.
LoggroupProps?	`logs.LogGroupProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o grupo de logs do CloudWatch Logs.

Propriedades de padrão

Nome	Tipo	Descrição
ApiGateway	`api.LambdaRestApi`	Retorna uma instância da API REST Gateway criada pelo padrão.
ApigateWayRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão para a API REST do API Gateway.
ApigatewayCloudWatchRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão que permite o log de acesso da API REST do API Gateway para o CloudWatch.
ApigatewayLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para o qual os logs de acesso à API REST do API Gateway são enviados.

Uso da API de amostra

Observações: Cada endpoint do SageMaker é exclusivo e a resposta da API dependerá do modelo implantado. O exemplo dado abaixo assume a amostra dePublicação no blog. Para obter uma referência sobre como isso seria implementado, consulteinteg.apigateway-sagemakerendpoint-overwrite.ts.

Método	O caminho da solicitação	Sequência de consulta	Ação SageMaker	Descrição
GET	`/321`	`items=101,131,162`	`sagemaker:InvokeEndpoint`	Recupera as previsões para um usuário e itens específicos.

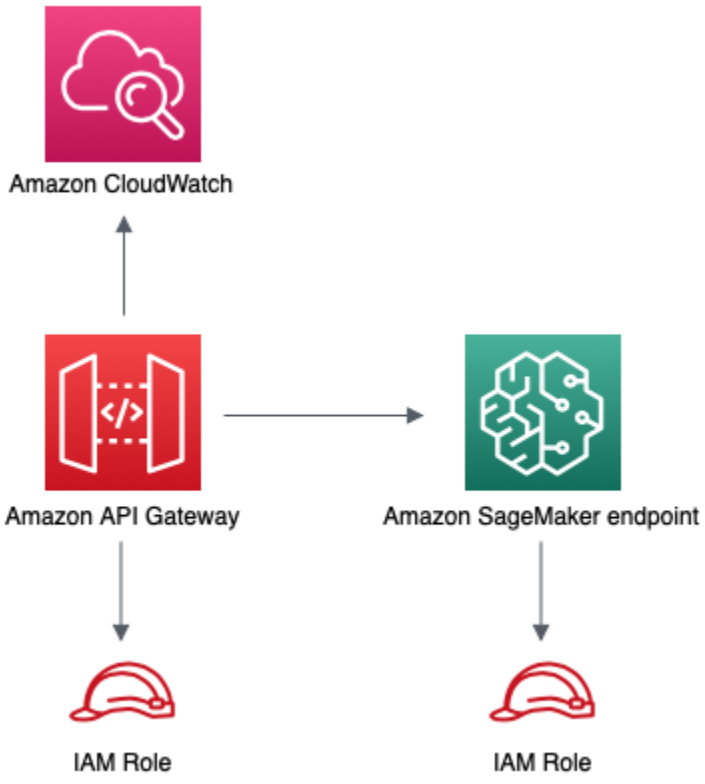
Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon API Gateway

- Implantar um endpoint de API otimizado para bordas
- Habilitar o log do CloudWatch para o
- Configurar a função IAM de acesso de menor privilégio para API Gateway
- Defina o AuthorizationType padrão para todos os métodos de API como IAM
- Ativar rastreamento do X-Ray
- Validar parâmetros de solicitação antes de passar dados para o SageMaker

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-apigateway-sagemakerendpoint

aws-apigateway-sqs

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àControle de versão semânticaModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_apigateway_sqs`
TypeScript	`@aws-solutions-constructs/aws-apigateway-sqs`
Java	`software.amazon.awsconstructs.services.apigatewaysqs`

Overview

Este AWS Solutions Construct implementa uma API REST do Amazon API Gateway conectada a uma fila do Amazon SQS.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { ApiGatewayToSqs, ApiGatewayToSqsProps } from "@aws-solutions-constructs/aws-apigateway-sqs";

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

Initializer

```
new ApiGatewayToSqs(scope: Construct, id: string, props: ApiGatewayToSqsProps);
```

Parâmetros

- escopo`Construct`

- `idstring`
- propsApiGatewayToSqsProps

Adereços de construção de padrão

Nome	Tipo	Descrição
ApigatewayProps?	`api.RestApiProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o API Gateway.
QueueProps?	`sqs.QueueProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão da fila.
Implementar Deadlette rQueue?	`boolean`	Se uma fila secundária deve ser usada como uma dead letter queue. Padronizado como `true`.
MaxReceiveCount	`number`	O número de vezes que uma mensagem pode ser desenfile irada sem sucesso antes de ser movida para a fila de mensagens mortas.
AllowCreateOperation?	`boolean`	Se deve implantar um Método API Gateway para criar operações na fila (ou seja, SQS:SendMessage).
CreateRequestTemplate?	`string`	Substitua o modelo de solicitação padrão do API Gateway para o método Criar, `seallowCreateOperati on` é definido como`true`.

Nome	Tipo	Descrição
AllowReadOperation?	`boolean`	Se deve implantar um Método API Gateway para operações de leitura na fila (ou seja, SQS:ReceiveMessage).
ReadRequestTemplate?	`string`	Substitua o modelo de solicitação padrão do API Gateway para o método de leitura, se `allowReadOperation` é definido como `true`.
AllowDeleteOperation?	`boolean`	Se deve implantar um Método API Gateway para operações Excluir na fila (por exemplo, SQS:DeleteMessage).
DeleteRequestTemplate?	`string`	Substitua o modelo de solicitação padrão do API Gateway para o método Delete, se `allowDeleteOperation` é definido como `true`.
LoggroupProps?	`logs.LogGroupProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão do grupo de logs do CloudWatch Logs.

Propriedades de padrão

Nome	Tipo	Descrição
APIGateway	`api.RestApi`	Retorna uma instância da API REST Gateway criada pelo padrão.
ApigatewayCloudWatchRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão que permite o log de acesso da API REST do API Gateway para o CloudWatch.
ApigatewayLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para o qual os logs de acesso à API REST do API Gateway são enviados.
ApigateWayRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão para a API REST do API Gateway.
DeadletterQueue?	`sqs.Queue`	Retorna uma instância da fila de letras mortas criada pelo padrão, se uma for implantad a.
SQSqueue	`sqs.Queue`	Retorna uma instância da fila SQS criada pelo padrão.

Uso da API de amostra

Método	Caminho da solicitação	Corpo da solicitação	Ação da Fila	Descrição
GET	`/`		`sqs::ReceiveMessage`	Recupera uma mensagem da fila.
POST	`/`	`{ "data": "Hello World!" }`	`sqs::SendMessage`	Entrega uma mensagem para a fila.
DELETE	`/message?receiptHandle=[value]`		`sqs::DeleteMessage`	Exclui uma mensagem especificada da fila

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

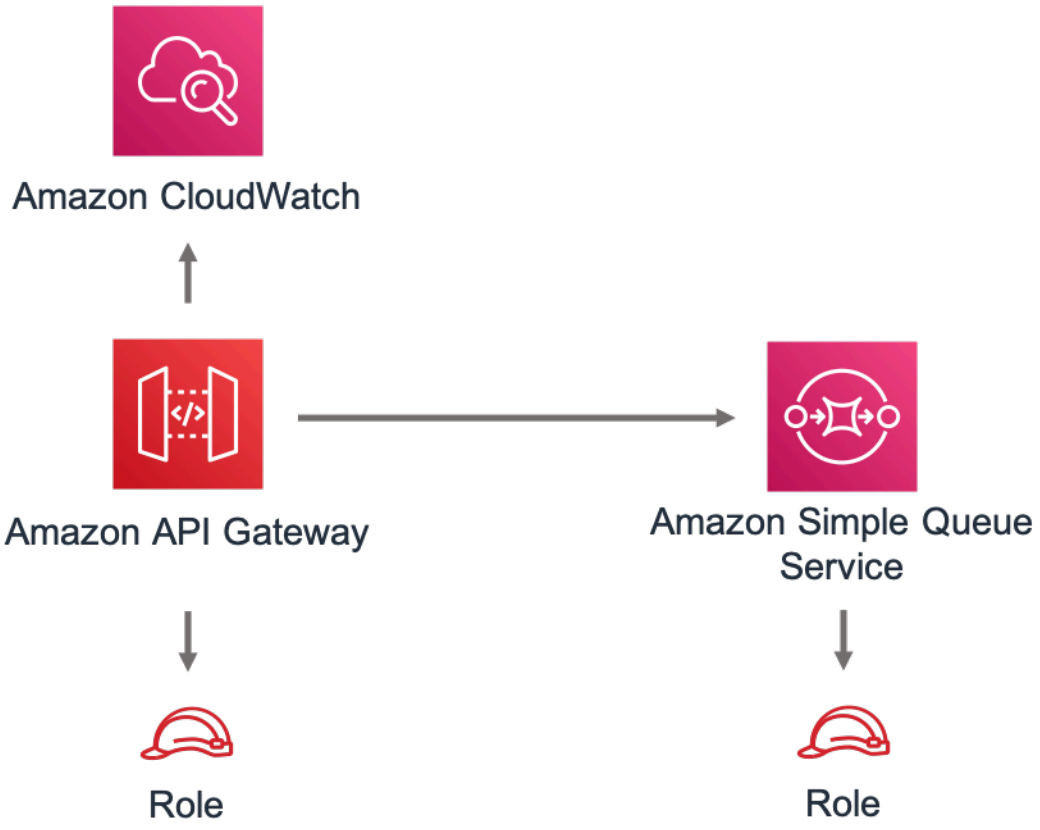
Amazon API Gateway

- Implantar um endpoint de API otimizado para bordas
- Habilitar o registro do CloudWatch para o
- Configurar a função IAM de acesso de menor privilégio para API Gateway
- Defina o AuthorizationType padrão para todos os métodos de API como IAM
- Habilitar rastreamento do X-Ray

Fila do Amazon SQS

- Implantar fila de mensagens mortas do SQS para a fila do SQS de origem
- Habilitar a criptografia do lado do servidor para a fila do SQS de origem AWS a Chave do KMS
- Aplique a criptografia de dados em trânsito

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

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

aws-cloudfront-apigateway

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_cloudfront_apigateway`
TypeScript	`@aws-solutions-constructs/aws-cloudfront-apigateway`
Java	`software.amazon.awsconstructs.services.cloudfrontapigateway`

Overview

Este AWS Solutions Construct implementa uma distribuição do Amazon CloudFront na frente de uma API REST do Amazon API Gateway.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import * as api from '@aws-cdk/aws-apigateway';
import * as lambda from "@aws-cdk/aws-lambda";
import { CloudFrontToApiGateway } from '@aws-solutions-constructs/aws-cloudfront-
apigateway';

const lambdaProps: lambda.FunctionProps = {
    code: lambda.Code.fromAsset(`${__dirname}/lambda`),
    runtime: lambda.Runtime.NODEJS_12_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
});
```

Initializer

```
new CloudFrontToApiGateway(scope: Construct, id: string, props:
 CloudFrontToApiGatewayProps);
```

Parâmetros

- escopoConstruct
- idstring
- propsCloudFrontToApiGatewayProps

Adereços de construção de padrão

Nome	Tipo	Descrição
ExistingApigatewayObj	api.RestApi	O API Gateway regional que será fornecido com o CloudFront
CloudFrontDistributionProps?	cloudfront.DistributionProps	O usuário opcional forneceu adereços para substituir

Nome	Tipo	Descrição
		os adereços padrão para a distribuição do CloudFront.
InserthttpSecurityHeaders?	`boolean`	O usuário opcional forneceu adereços para ativar/de sativar a injeção automática de cabeçalhos de segurança HTTP de práticas recomenda das em todas as respostas do CloudFront

Propriedades de padrão

Nome	Tipo	Descrição
APIGateway	`api.RestApi`	Retorna uma instância da API REST Gateway criada pelo padrão.
CloudFrontLoggingBucket?	`s3.Bucket`	Retorna uma instância do bucket de log criado pelo padrão para a distribuição da Web do CloudFront.
CloudFrontWebDistribution	`cloudfront.CloudFrontWebDistribution`	Retorna uma instância da distribuição web do CloudFron t criada pelo padrão.
EdgeLambdaFunction Version?	`lambda.Version`	Retorna uma instância da versão da função de borda do Lambda criada pelo padrão.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

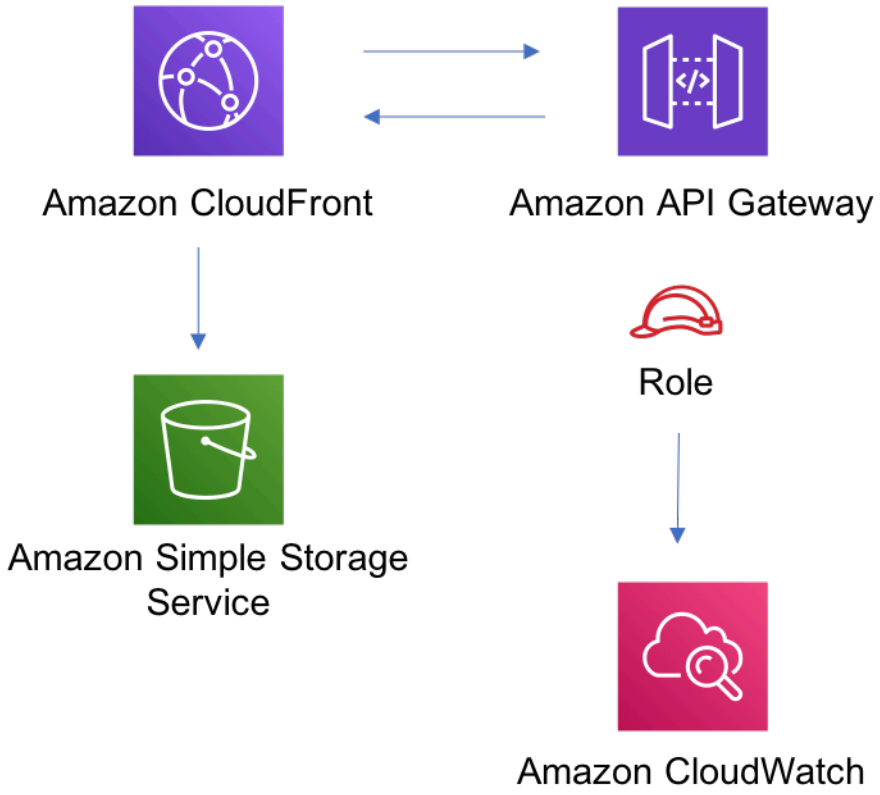
Amazon CloudFront

- Configurar registro de acesso para o CloudFront WebDistribution
- Habilite a injeção automática de cabeçalhos de segurança HTTP de práticas recomendadas em todas as respostas do CloudFront WebDistribution

Amazon API Gateway

- O objeto API Gateway fornecido pelo usuário é usado como está
- Ativar rastreamento do X-Ray

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-cloudfront-apigateway

aws-cloudfront-apigateway-lambda

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_cloudfront_apigateway_lambda`
TypeScript	`@aws-solutions-constructs/aws-cloudfront-apigateway-lambda`
Java	`software.amazon.awsconstructs.services.cloudfrontapigatewaylambda`

Overview

Este AWS Solutions Construct implementa uma distribuição do Amazon CloudFront na frente de uma API REST suportada pelo Amazon API Gateway Lambda.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { CloudFrontToApiGatewayToLambda } from '@aws-solutions-constructs/aws-
cloudfront-apigateway-lambda';

new CloudFrontToApiGatewayToLambda(this, 'test-cloudfront-apigateway-lambda', {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    }
});
```

Initializer

```
new CloudFrontToApiGatewayToLambda(scope: Construct, id: string, props:
 CloudFrontToApiGatewayToLambdaProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`CloudFrontToApiGatewayToLambdaProps`

Adereços de construção de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo o tanto isso `elambdaFunctionProps` causará um erro.

Nome	Tipo	Descrição
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.
ApigatewayProps?	`api.LambdaRestApiProps`	Os adereços fornecidos pelo usuário opcionais para substituir os adereços padrão do API Gateway
CloudFrontDistributionProps?	`cloudfront.DistributionProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para a distribuição do CloudFront.
InserthttpSecurityHeaders?	`boolean`	O usuário opcional forneceu adereços para ativar/de sativar a injeção automática de cabeçalhos de segurança HTTP de práticas recomenda das em todas as respostas do CloudFront
LoggroupProps?	`logs.LogGroupProps`	Props fornecidos pelo usuário para substituir os adereços padrão para o grupo de logs do CloudWatch Logs.

Propriedades de padrão

Nome	Tipo	Descrição
ApiGateway	`api.RestApi`	Retorna uma instância da API REST Gateway criada pelo padrão.
ApigatewayCloudWatchRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão que permite o log de acesso da API REST do API Gateway para o CloudWatch.
ApigatewayLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para o qual os logs de acesso à API REST do API Gateway são enviados.
CloudFrontLoggingBucket?	`s3.Bucket`	Retorna uma instância do bucket de log criado pelo padrão para a distribuição da Web do CloudFront.
CloudFrontWebDistribution	`cloudfront.CloudFrontWebDistribution`	Retorna uma instância da distribuição web do CloudFront criada pelo padrão.
EdgeLambdaFunctionVersion?	`lambda.Version`	Retorna uma instância da versão da função de borda do Lambda criada pelo padrão.
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon CloudFront

- Configurar registro de acesso para o CloudFront WebDistribution
- Habilite a injeção automática de cabeçalhos de segurança HTTP de práticas recomendadas em todas as respostas do CloudFront WebDistribution

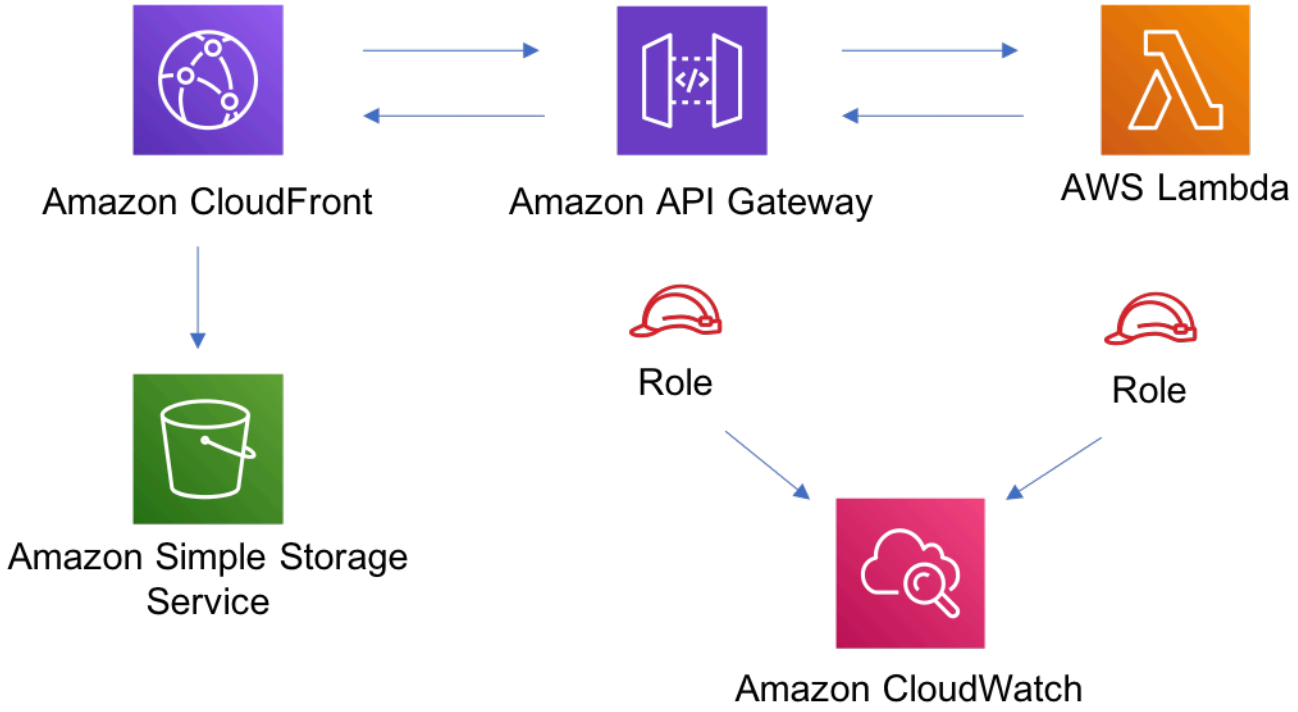
Amazon API Gateway

- Implantar um ponto final de API regional
- Habilitar o log do CloudWatch para o
- Configurar a função IAM de acesso de menor privilégio para API Gateway
- Defina o AuthorizationType padrão para todos os métodos de API como IAM
- Ativar rastreamento do X-Ray

Função do AWS Lambda

- Configuração da função do IAM de acesso de privilégio limitado para a função Lambda
- Ativar a reutilização de conexões com a função Keep-Alive para NodeJS Lambda
- Ativar rastreamento do X-Ray
- Definição de variáveis de ambiente:
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-cloudfront-apigateway-lambda

aws-cloudfront-mediastore

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

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

Overview

Este AWS Solutions Construct implementa uma distribuição do Amazon CloudFront conectada a um contêiner AWS Elemental MediaStore.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { CloudFrontToMediaStore } from '@aws-solutions-constructs/aws-cloudfront-mediastore';

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

Initializer

```
new CloudFrontToMediaStore(scope: Construct, id: string, props: CloudFrontToMediaStoreProps);
```

Parâmetros

- escopo`Construct`
- id`string`

- propsCloudFrontToMediaStoreProps

Props de criação de padrão

Nome	Tipo	Descrição
ExistingMediaStoreContainerObj?	mediastore.CfnContainer	Recipiente MediaStore opcional fornecido pelo usuário para substituir o contêiner MediaStore padrão.
MediaStoreContainerProps?	mediastore.CfnContainerProps	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o Container MediaStore.
CloudFrontDistributionProps?	cloudfront.DistributionProps | any	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para a distribuição do CloudFront.
InserthttpSecurityHeaders?	boolean	Props opcionais fornecidos pelo usuário para ativar/desativar a injeção automática de cabeçalhos de segurança HTTP de práticas recomendadas em todas as respostas do CloudFront.

Propriedades de padrão

Nome	Tipo	Descrição
CloudFrontWebDistribution	cloudfront.CloudFrontWebDistribution	Retorna uma instância da distribuição web do CloudFront criada pelo padrão.

Nome	Tipo	Descrição
MediaStoreContainer	`mediastore.CfnContainer`	Retorna uma instância do contêiner MediaStore criado pelo padrão.
CloudFrontLoggingBucket	`s3.Bucket`	Retorna uma instância do bucket de log criado pelo padrão para a distribuição da Web do CloudFront.
CloudFrontOriginRequestPolicy	`cloudfront.OriginRequestPolicy`	Retorna uma instância da política de solicitação de origem do CloudFront criada pelo padrão para a distribuição da Web do CloudFront.
CloudFrontOriginAccessIdentity?	`cloudfront.OriginAccessIdentity`	Retorna uma instância da identidade de acesso de origem do CloudFront criada pelo padrão para a distribuição da Web do CloudFront.
EdgeLambdaFunctionVersion	`lambda.Version`	Retorna uma instância da versão da função de borda do Lambda criada pelo padrão.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

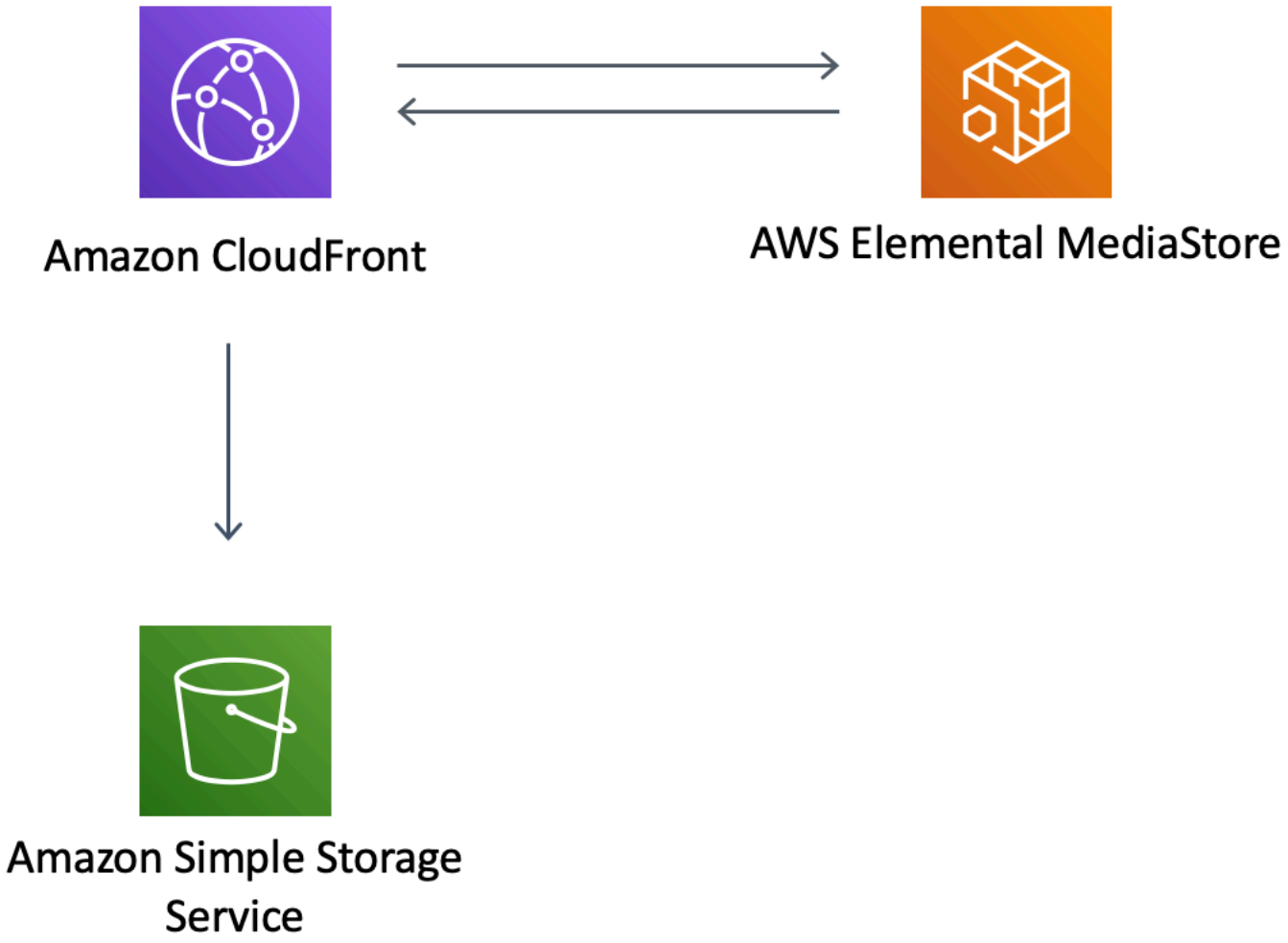
Amazon CloudFront

- Configurar o registro de acesso para a distribuição da Web CloudFront
- Ativar política de solicitação de origem do CloudFront para contêiner AWS Elemental MediaStore
- Defina`User-Agent`cabeçalho personalizado com identidade de acesso de origem do CloudFront
- Habilite a injeção automática de cabeçalhos de segurança HTTP de práticas recomendadas em todas as respostas da distribuição na Web do CloudFront

AWS Elemental MediaStore

- Definir a política de exclusão para reter o recurso
- Definir o nome do contêiner com o nome da pilha do CloudFormation
- Definir o padrãoPolítica de compartilhamento de recursos de origem cruzada (CORS) de contêiner
- Definir o padrãoPolítica de ciclo de vida
- Definir o padrãoPolítica de contêinerpara permitir apenas`aws:UserAgent`com origem no CloudFront de origem
- Definir o padrãoPolítica de métrica
- Habilitar registro em log de acesso

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-cloudfront-mediastore

aws-cloudfront-s3

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àSemantic versionModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_cloudfront_s3`
TypeScript	`@aws-solutions-constructs/aws-cloudfront-s3`
Java	`software.amazon.awsconstructs.services.cloudfronts3`

Overview

Este AWS Solutions Construct implementa uma distribuição do Amazon CloudFront na frente de um bucket do Amazon S3.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { CloudFrontToS3 } from '@aws-solutions-constructs/aws-cloudfront-s3';

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

Initializer

```
new CloudFrontToS3(scope: Construct, id: string, props: CloudFrontToS3Props);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`CloudFrontToS3Props`

Props de criação de padrão

Nome	Tipo	Descrição
ExistingBucketoBJ?	`s3.Bucket`	Instância existente do objeto S3 Bucket. Se isso for fornecido, então também fornecendo`bucketProps` é um erro.
Baldes?	`s3.BucketProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão do bucket. Ignorado se um`existingBucketObj` é fornecido.

Nome	Tipo	Descrição
CloudFrontDistributionProps?	`cloudfront.DistributionProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para a distribuição do CloudFront.
InserthttpSecurityHeaders?	`boolean`	O usuário opcional forneceu adereços para ativar/desativar a injeção automática de cabeçalhos de segurança HTTP de práticas recomendadas em todas as respostas do CloudFront

Propriedades do padrão

Nome	Tipo	Descrição
CloudFrontWebDistribution	`cloudfront.CloudFrontWebDistribution`	Retorna uma instância da distribuição web do CloudFront criada pelo padrão.
S3 Bucket?	`s3.Bucket`	Retorna uma instância do bucket S3 criado pelo padrão.
S3loggingBucket?	`s3.Bucket`	Retorna uma instância do bucket de log criado pelo padrão para o bucket S3.
EdgeLambdaFunctionVersion?	`lambda.Version`	Retorna uma instância da versão da função de borda do Lambda criada pelo padrão.
CloudFrontLoggingBucket?	`s3.Bucket`	Retorna uma instância do bucket de log criado pelo

Nome	Tipo	Descrição
		padrão para a distribuição da Web do CloudFront.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

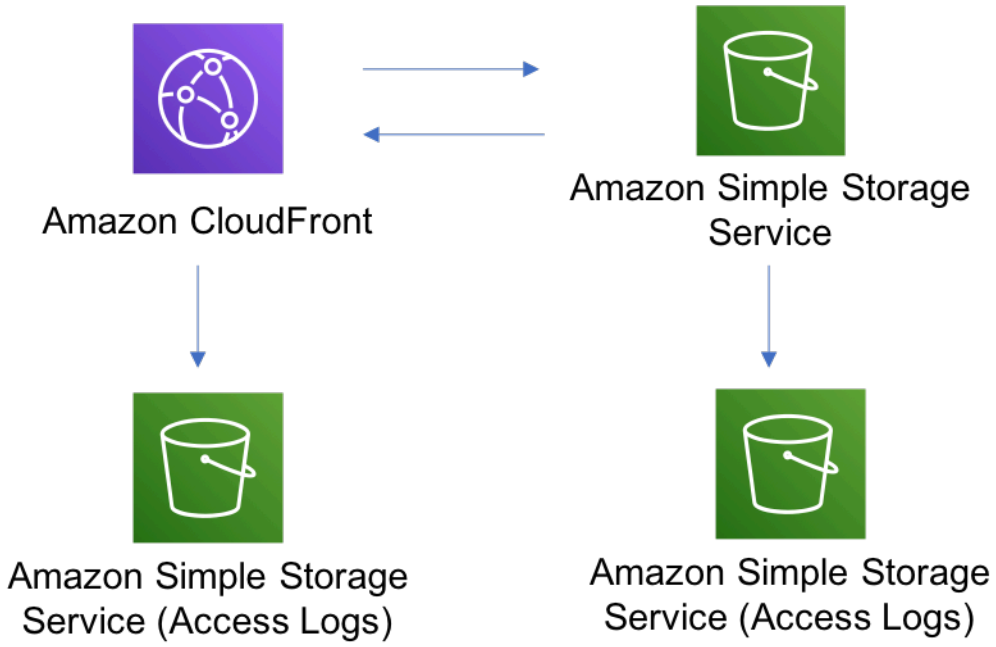
Amazon CloudFront

- Configurar registro de acesso para o CloudFront WebDistribution
- Habilite a injeção automática de cabeçalhos de segurança HTTP de práticas recomendadas em todas as respostas do CloudFront WebDistribution

Amazon S3 Bucket

- Configurar registro de acesso para o bucket do S3
- Ativar criptografia no lado do servidor para o S3 Bucket usando a chave KMS gerenciada pela AWS
- Ativar o controle de versão para o bucket do S3
- Não permitir acesso público para o S3 Bucket
- Manter o bucket do S3 ao excluir a pilha do CloudFormation
- Aplique a criptografia de dados em trânsito
- Aplica regra de ciclo de vida para mover versões de objetos não atuais para o armazenamento do Glacier após 90 dias

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

	@aws -solutions-constructs/aws-cloudfront-s3

aws-cognito-apigateway-lambda

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_cognito_apigateway_lambda`
TypeScript	`@aws-solutions-constructs/aws-cognito-apigateway-lambda`
Java	`software.amazon.awsconstructs.services.cognitoapigatewaylambda`

Overview

Este AWS Solutions Construct implementa o Amazon Cognito protegendo uma API REST apoiada pelo Amazon API Gateway Lambda.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { CognitoToApiGatewayToLambda } from '@aws-solutions-constructs/aws-cognito-apigateway-lambda';

new CognitoToApiGatewayToLambda(this, 'test-cognito-apigateway-lambda', {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    }
});
```

Se você estiver definindo recursos e métodos em sua API (por exemplo,`proxy = false`), você deve chamar o`addAuthorizers()`depois que a API é totalmente definida. Isso garante que todos os métodos em sua API estejam protegidos.

Veja um exemplo em TypeScript:

```
import { CognitoToApiGatewayToLambda } from '@aws-solutions-constructs/aws-cognito-
apigateway-lambda';

const construct = new CognitoToApiGatewayToLambda(this, 'test-cognito-apigateway-
lambda', {
    lambdaFunctionProps: {
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        runtime: lambda.Runtime.NODEJS_12_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();
```

Initializer

```
new CognitoToApiGatewayToLambda(scope: Construct, id: string, props:
 CognitoToApiGatewayToLambdaProps);
```

Parâmetros

- escopoConstruct
- idstring
- propsCognitoToApiGatewayToLambdaProps

Adereços de construção de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo tanto isso `elambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.
ApigatewayProps?	`api.LambdaRestApiProps`	Os adereços fornecidos pelo usuário opcionais para substituir os adereços padrão do API Gateway
CognitouserPoolProps?	`cognito.UserPoolProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para o grupo de usuários do Cognito
CognitouserPoolClientProps?	`cognito.UserPoolClientProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para o cliente do grupo de usuários do Cognito
LoggroupProps?	`logs.LogGroupProps`	Aderos opcionais fornecidos pelo usuário para substituir os adereços padrão para o grupo de logs do CloudWatch Logs.

Propriedades de padrão

Nome	Tipo	Descrição
ApiGateway	`api.RestApi`	Retorna uma instância da API REST Gateway criada pelo padrão.
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.
userPool	`cognito.UserPool`	Retorna uma instância do grupo de usuários do Cognito criado pelo padrão.
USPoolClient	`cognito.UserPoolClient`	Retorna uma instância do cliente do grupo de usuários do Cognito criado pelo padrão.
ApigatewayCloudWatchRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão que permite o log de acesso da API REST do API Gateway para o CloudWatch.
ApigatewayLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para o qual os logs de acesso à API REST do API Gateway são enviados.
ApigatewayAuthorizer	`api.CfnAuthorizer`	Retorna uma instância do autorizador API Gateway criado pelo padrão.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon Cognito

- Definição de política de senha para grupos de usuários
- Impor o modo de segurança avançado para grupos de usuários

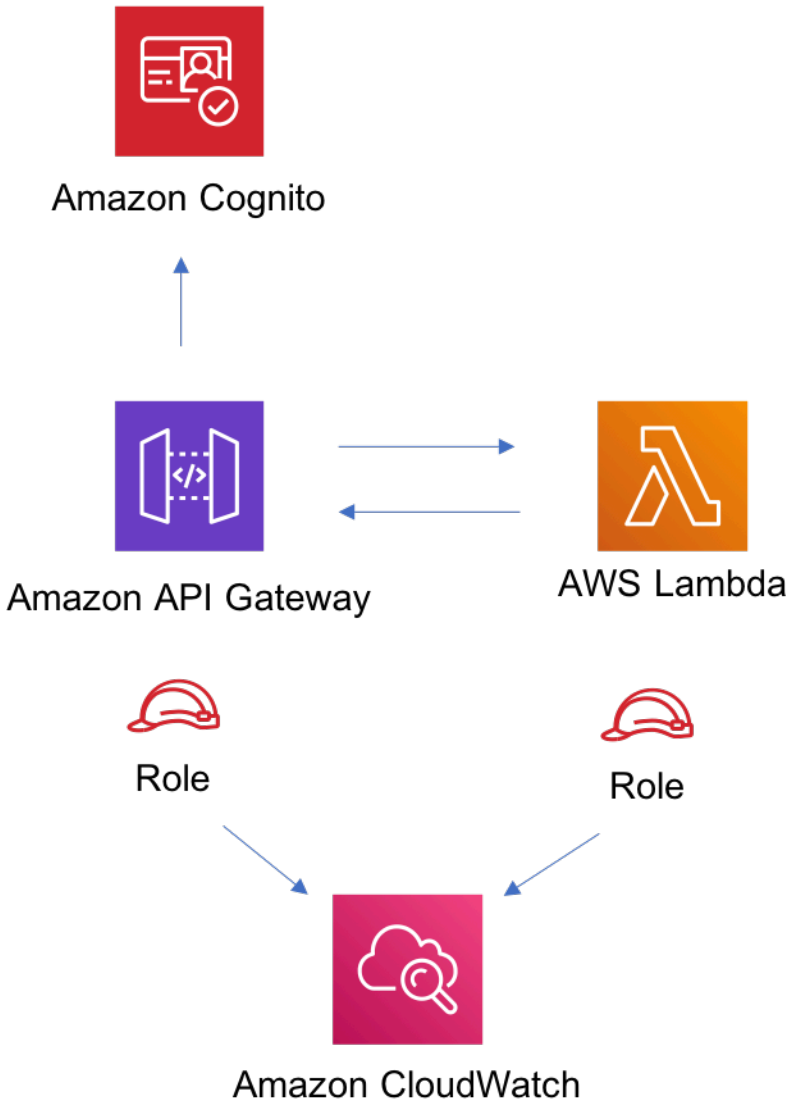
Amazon API Gateway

- Implantar um endpoint de API otimizado para bordas
- Habilitar o log do CloudWatch para o
- Configurar a função IAM de acesso de menor privilégio para API Gateway
- Defina o AuthorizationType padrão para todos os métodos de API como IAM
- Ativar rastreamento do X-Ray

Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Ativar a reutilização de conexões com a função Keep-Alive para NodeJS Lambda
- Ativar rastreamento do X-Ray
- Definição de variáveis de ambiente:
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

	[@aws -solutions-constructs/aws-cognito-apigateway-lambda]()

aws-dynamodb-stream-lambda

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_dynamodb_stream_lambda`
TypeScript	`@aws-solutions-constructs/aws-dynamodb-stream-lambda`
Java	`software.amazon.awsconstructs.services.dynamodbstreamlambda`

Overview

Este AWS Solutions Construct implementa uma tabela padrão do Amazon DynamoDB com stream para invocar a função do AWS Lambda com as permissões menos privilegiadas.

Aqui está uma definição de padrão implantável mínima:

```
import { DynamoDBStreamToLambdaProps, DynamoDBStreamToLambda} from '@aws-solutions-constructs/aws-dynamodb-stream-lambda';

new DynamoDBStreamToLambda(this, 'test-dynamodb-stream-lambda', {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    },
```

```
});
```

Initializer

```
new DynamoDBStreamToLambda(scope: Construct, id: string, props:
 DynamoDBStreamToLambdaProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`DynamoDBStreamToLambdaProps`

Adereços de criação de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo tanto isso `elambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.
DynamotableProps?	`dynamodb.TableProps`	Os adereços fornecidos pelo usuário opcionais para substituir os adereços padrão da Tabela do DynamoDB

Nome	Tipo	Descrição
ExistingTableobj?	`dynamodb.Table`	Instância existente do objeto de tabela do DynamoDB, fornecendo tanto isso quanto`dynamoTableProps` causará um erro.
DynamoEventSourceProps?	`aws-lambda-event-sources.DynamoEventSourceProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para a Origem do evento do DynamoDB

Propriedades do padrão

Nome	Tipo	Descrição
DynamoTable	`dynamodb.Table`	Retorna uma instância da tabela do DynamoDB criada pelo padrão.
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.

Função Lambda

Esse padrão requer uma função do Lambda que possa postar dados no serviço Elasticsearch a partir do stream do DynamoDB. Uma função de exemplo é fornecidaAqui.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

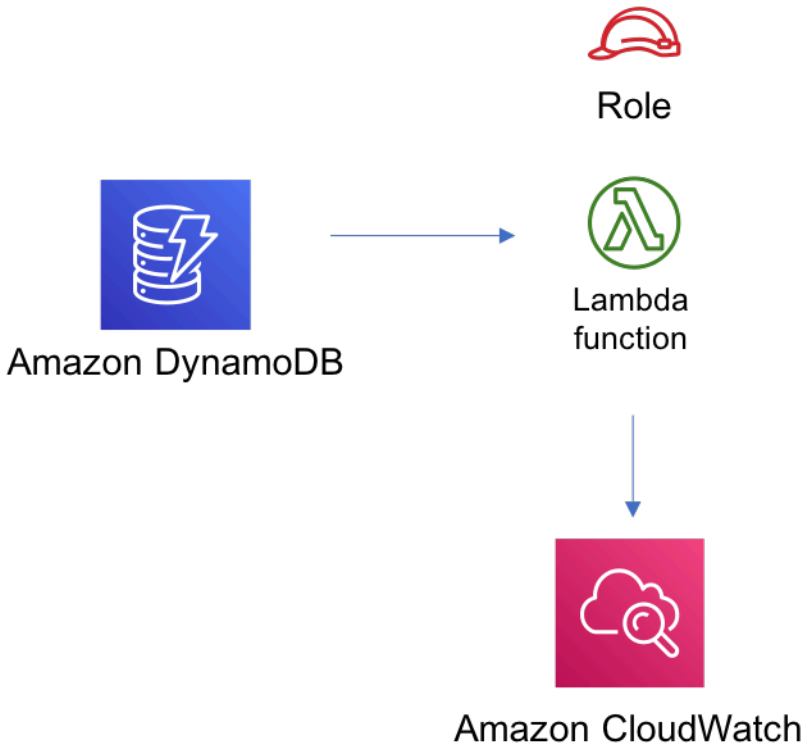
Amazon DynamoDB Tabela

- Definir o modo de faturamento da Tabela do DynamoDB como On-Demand (Pagamento por solicitação)
- Habilitar criptografia no lado do servidor para a tabela do DynamoDB usando a chave KMS gerenciada pela AWS
- Cria uma chave de partição chamada 'id' para a tabela do DynamoDB
- Manter a tabela ao excluir a pilha do CloudFormation
- Permita backups contínuos e recuperação point-in-time

Função do AWS Lambda

- Configuração da função do IAM de acesso de privilégio limitado para Lambda
- Ativar a reutilização de conexões com a função Keep-Alive para NodeJS Lambda
- Ativar rastreamento do X-Ray
- Ativar recursos de tratamento de falhas: habilitar bisect na função Erro; definir a Idade Máxima de Registro padrão (24 horas); definir Máximo de Tentativas de Repetição (500) padrão; e implantar a fila de letras mortas SQS como destino em caso de falha
- Definir variáveis de ambiente:
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

	@aws -solutions-constructs/aws-dynamodb-stream-lambda

aws-dynamodb-stream-lambda-elasticsearch-kibana

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_dynamodb_stream_lambda_elasticsearch_kibana`
TypeScript	`@aws-solutions-constructs/aws-dynamodb-stream-lambda-elasticsearch-kibana`
Java	`software.amazon.awsconstructs.services.dynamodbstreamlambdaelasticsearchkibana`

Overview

Este AWS Solutions Construct implementa a tabela do Amazon DynamoDB com stream, uma função do AWS Lambda e um Amazon Elasticsearch Service com as permissões menos privilegiadas.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { DynamoDBStreamToLambdaToElasticSearchAndKibana,
 DynamoDBStreamToLambdaToElasticSearchAndKibanaProps } from '@aws-solutions-constructs/
aws-dynamodb-stream-lambda-elasticsearch-kibana';
import { Aws } from "@aws-cdk/core";

const props: DynamoDBStreamToLambdaToElasticSearchAndKibanaProps = {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    },
    domainName: 'test-domain',
    // TODO: Ensure the Cognito domain name is globally unique
    cognitoDomainName: 'globallyuniquedomain' + Aws.ACCOUNT_ID;
};
```

```
new DynamoDBStreamToLambdaToElasticSearchAndKibana(this, 'test-dynamodb-stream-lambda-elasticsearch-kibana', props);
```

Initializer

```
new DynamoDBStreamToLambdaToElasticSearchAndKibana(scope: Construct, id: string, props: DynamoDBStreamToLambdaToElasticSearchAndKibanaProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`DynamoDBStreamToLambdaToElasticSearchAndKibanaProps`

Props de criação de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo tanto isso `elambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.
DynamotableProps?	`dynamodb.TableProps`	Os adereços fornecidos pelo usuário opcionais para

Nome	Tipo	Descrição
		substituir os adereços padrão da Tabela do DynamoDB
ExistingTableobj?	`dynamodb.Table`	Instância existente do objeto de tabela do DynamoDB, fornecendo tanto isso quanto `dynamoTableProps` causará um erro.
DynamoEventSourceProps?	`aws-lambda-event-sources.DynamoEventSourceProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para a Origem do evento do DynamoDB
EsdomainProps?	`elasticsearch.CfnDomainProps`	O usuário opcional forneceu adereços para substituir os adereços padrão do Amazon Elasticsearch Service
domainName	`string`	Nome de domínio para o Cognito e o Amazon Elasticsearch Service
CreateCloudWatchAlms	`boolean`	Criar alarmes recomendados do CloudWatch.

Propriedades do padrão

Nome	Tipo	Descrição
CloudwatchAlarm?	`cloudwatch.Alarm[]`	Retorna uma lista de um ou mais alarmes do CloudWatch criados pelo padrão.

Nome	Tipo	Descrição
DynamoTable	dynamodb.Table	Retorna uma instância da tabela do DynamoDB criada pelo padrão.
ElasticSearchDomain	elasticsearch.CfnDomain	Retorna uma instância do domínio Elasticsearch criado pelo padrão.
IdentityPool	cognito.CfnIdentityPool	Retorna uma instância do pool de identidades do Cognito criado pelo padrão.
LambdaFunction	lambda.Function	Retorna uma instância da função Lambda criada pelo padrão.
userPool	cognito.UserPool	Retorna uma instância do grupo de usuários do Cognito criado pelo padrão.
USPoolClient	cognito.UserPoolClient	Retorna uma instância do cliente do grupo de usuários do Cognito criado pelo padrão.

Função Lambda

Esse padrão requer uma função do Lambda que possa postar dados no serviço Elasticsearch a partir do stream do DynamoDB. Uma função de exemplo é fornecidaAqui.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Tabela do Amazon DynamoDB

- Definir o modo de faturamento da Tabela do DynamoDB como On-Demand (Pagamento por solicitação)

- Habilitar criptografia no lado do servidor para a tabela do DynamoDB usando a chave KMS gerenciada pela AWS
- Cria uma chave de partição chamada 'id' para a tabela do DynamoDB
- Manter a tabela ao excluir a pilha do CloudFormation
- Habilita backups contínuos e recuperação point-in-time

Função do AWS Lambda

- Configuração da função do IAM de acesso a privilégio limitado para função do Lambda
- Ativar a reutilização de conexões com a função Keep-Alive para NodeJS Lambda
- Habilitar rastreamento do X-Ray
- Ativar recursos de tratamento de falhas: habilitar bisect na função Erro; definir a Idade Máxima de Registro padrão (24 horas); definir Máximo de Tentativas de Repetição (500) padrão; e implantar a fila de letras mortas SQS como destino em caso de falha
- SET variáveis de ambiente:
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

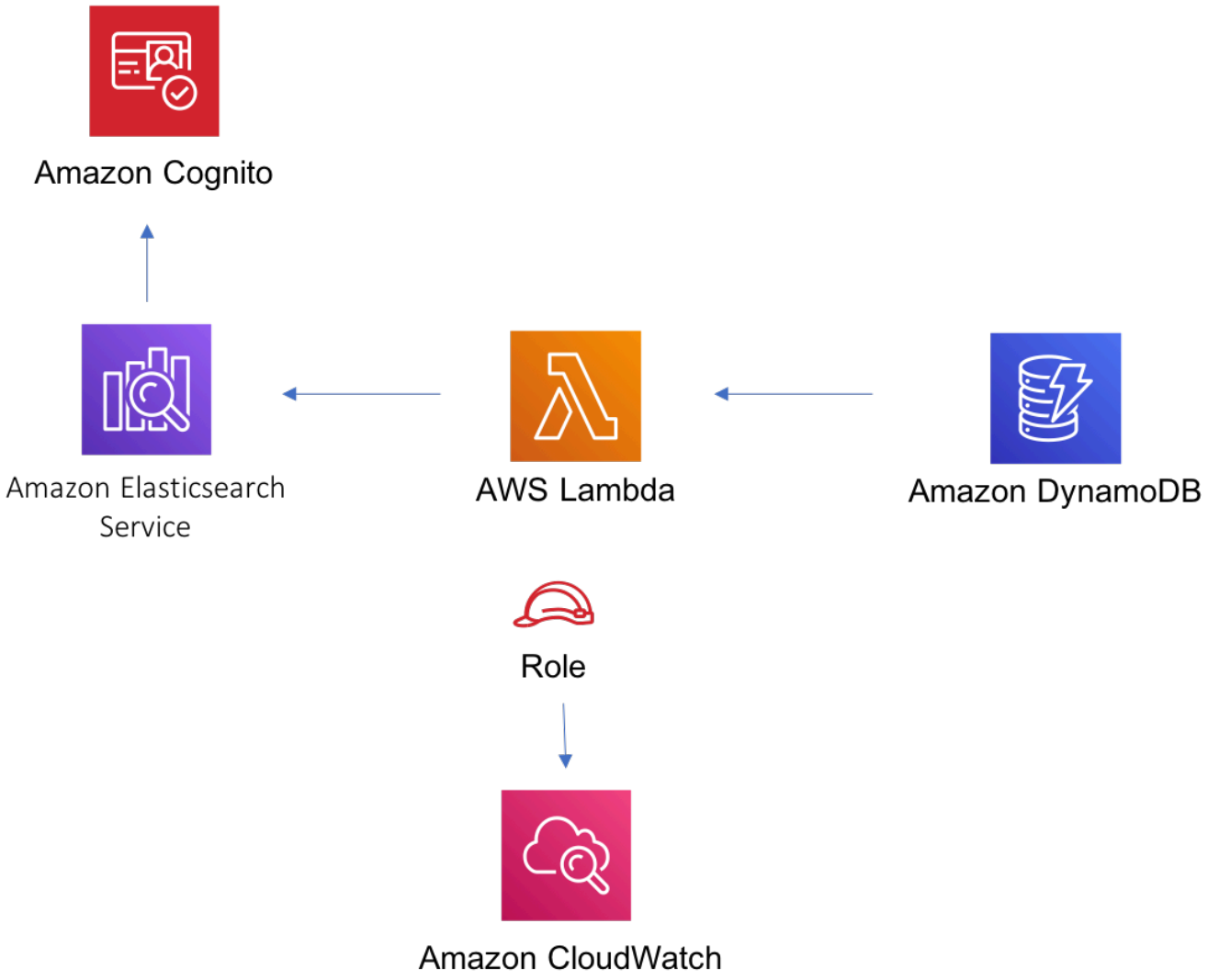
Amazon Cognito

- Definir política de senha para grupos de usuários
- Impor o modo de segurança avançado para grupos de usuários

Amazon Elasticsearch Service

- Implantar as melhores práticas Alarmes do CloudWatch para o domínio do Elasticsearch
- Proteja o acesso ao painel do Kibana com grupos de usuários do Cognito
- Ativar criptografia no lado do servidor para o Elasticsearch Domain usando a chave KMS gerenciada pela AWS
- Habilitar a criptografia de nó a nó para o domínio do Elasticsearch
- Configuração do cluster para o domínio do Amazon ES

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-dynamodb-stream-lambda-elasticsearch-kibana

aws-events-rule-kinesisfirehose-s3

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àControle de versão semânticaModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_events_rule_kinesisfirehose_s3`
TypeScript	`@aws-solutions-constructs/aws-events-rule-kinesisfirehose-s3`
Java	`software.amazon.awsconstructs.services.eventsrulekinesisfirehoses3`

Overview

Este AWS Solutions Construct implementa uma regra do Amazon CloudWatch Events para enviar dados para um stream de distribuição do Amazon Kinesis Data Firehose conectado a um bucket do Amazon S3.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import * as cdk from '@aws-cdk/core';
import { EventsRuleToKinesisFirehoseToS3, EventsRuleToKinesisFirehoseToS3Props } from
 '@aws-solutions-constructs/aws-events-rule-kinesisfirehose-s3';

const eventsRuleToKinesisFirehoseToS3Props: EventsRuleToKinesisFirehoseToS3Props = {
    eventRuleProps: {
    schedule: events.Schedule.rate(cdk.Duration.minutes(5))
    }
```

```
};

new EventsRuleToKinesisFirehoseToS3(this, 'test-events-rule-firehose-s3',
 eventsRuleToKinesisFirehoseToS3Props);
```

Initializer

```
new EventsRuleToKinesisFirehoseToS3(scope: Construct, id: string, props:
 EventsRuleToKinesisFirehoseToS3Props);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`EventsRuleToKinesisFirehoseToS3Props`

Estrutura de padrão

Nome	Tipo	Descrição
EventruleProps	`events.RuleProps`	Propriedades fornecidas pelo usuário para substituir as propriedades padrão da regra CloudWatch Events.
Kinesis é Firehoseprops?	`aws-kinesisfirehose.CfnDeliveryStreamProps`	O usuário opcional forneceu adereços para substituir os adereços padrão do Kinesis Firehose Delivery Stream.
ExistingBucketoBJ?	`s3.IBucket`	Instância existente do objeto S3 Bucket. Se isso for fornecido, então também fornecendo`bucketProps` é um erro.

Nome	Tipo	Descrição
Baldes?	s3.BucketProps	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o bucket S3.
LoggroupProps?	logs.LogGroupProps	Props opcionais fornecidos pelo usuário para substituir os adereços padrão do grupo de logs do CloudWatch Logs.

Propriedades de padrão

Nome	Tipo	Descrição
EventsRule	events.Rule	Retorna uma instância da regra Eventos criada pelo padrão.
KinesisFirehose	kinesisfirehose.CfnDeliveryStream	Retorna uma instância do stream de entrega do Kinesis Firehose criado pelo padrão.
S3 Bucket	s3.Bucket	Retorna uma instância do bucket S3 criado pelo padrão.
S3loggingBucket?	s3.Bucket	Retorna uma instância do bucket de log criado pelo padrão para o bucket S3.
Função de eventos?	iam.Role	Retorna uma instância da função criada pela construçã o para a regra CloudWatch Events.

Nome	Tipo	Descrição
KinesisFireHoserole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão para o stream de entrega do Kinesis Firehose.
KinesisFireHoseLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para o qual os logs de acesso do Kinesis Firehose são enviados.

Configuração padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Regras de Amazon CloudWatch Events

- Configure a função do IAM de acesso de menor privilégio para a Regra de Eventos publicar no Kinesis Firehose Delivery Stream.

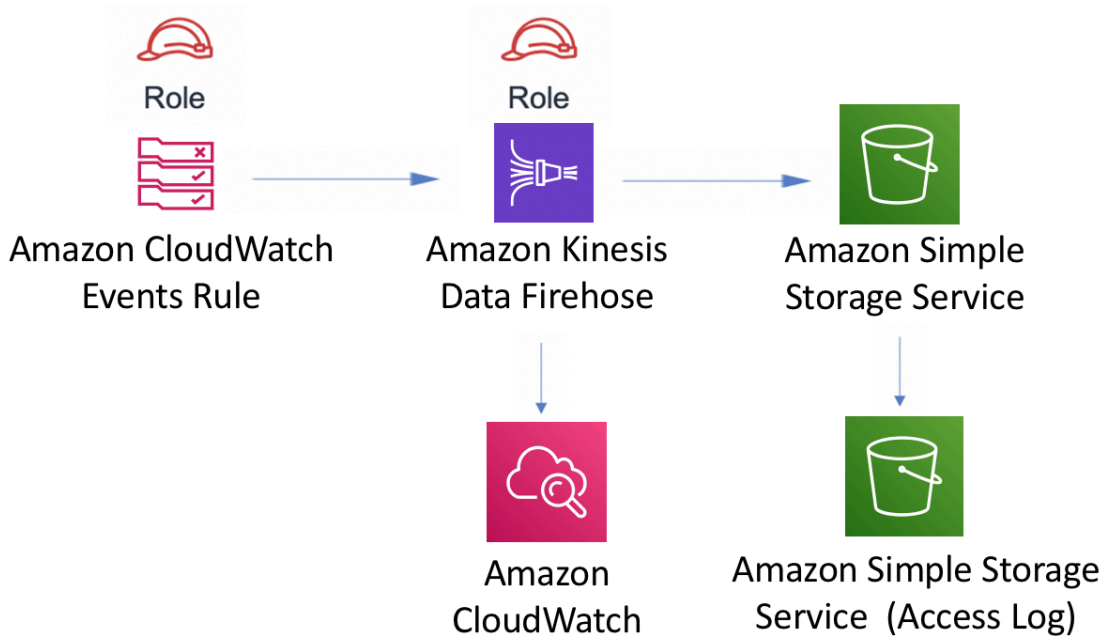
Amazon Kinesis Firehose

- Ative o registro do CloudWatch para o Kinesis Firehose.
- Configurar a função do IAM de acesso de menor privilégio para o Amazon Kinesis Firehose.

Bucket do Amazon S3

- Configurar o log de acesso para bucket.
- Habilite a criptografia do lado do servidor para bucket usando a chave KMS gerenciada pela AWS.
- Ative o controle de versão do bucket.
- Não permitir acesso público para o bucket.
- Mantenha o bucket ao excluir a pilha do CloudFormation.
- Aplica a regra de ciclo de vida para mover versões de objetos não atuais para o armazenamento do Glacier após 90 dias.

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-events-rule-kinesisfirehose-s3

aws-events-rule-kinesisstreams

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_events_rule_kinesisstream`
TypeScript	`@aws-solutions-constructs/aws-events-rule-kinesisstreams`
Java	`software.amazon.awsconstructs.services.eventsrulekinesisstream`

Overview

Este AWS Solutions Construct implementa uma regra de Amazon CloudWatch Events para enviar dados para um stream de dados do Amazon Kinesis.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import * as cdk from '@aws-cdk/core';
import {EventsRuleToKinesisStreams, EventsRuleToKinesisStreamsProps} from "@aws-solutions-constructs/aws-events-rule-kinesisstreams";

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

new EventsRuleToKinesisStreams(this, 'test-events-rule-kinesis-stream', props);
```

Initializer

```
new EventsRuleToKinesisStreams(scope: Construct, id: string, props:
 EventsRuleToKinesisStreamsProps);
```

Parâmetros

- escopoConstruct
- idstring
- propsEventsRuleToKinesisStreamsProps

Estrutura de padrão

Nome	Tipo	Descrição
EventruleProps	events.RuleProps	Propriedades fornecidas pelo usuário para substituir as propriedades padrão da regra CloudWatch Events.
ExistingStreamobj?	kinesis.Stream	Instância existente do Kinesis Stream, fornecendo tanto isso quantokinesisStreamProps causará um erro.
KinesisStreamprops?	kinesis.StreamProps	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o stream do Kinesis.
CreateCloudWatchAlms	boolean	Criar alarmes recomendados do CloudWatch.

Propriedades de padrão

Nome	Tipo	Descrição
EventsRule	`events.Rule`	Retorna uma instância da regra Eventos criada pelo padrão.
KinesisStream	`kinesis.Stream`	Retorna uma instância do stream do Kinesis criado pelo padrão.
Função de eventos?	`iam.Role`	Retorna uma instância da função criada pela construçã o para a regra CloudWatch Events.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon CloudWatch Events

- Configure a função do IAM de acesso de menor privilégio para a Regra de Eventos publicar no Kinesis Data Stream.

Amazon Kinesis Stream

- Ative a criptografia do lado do servidor para o Kinesis Data Stream usando a chave KMS gerenciada da AWS.

Architecture

GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

	@aws -solutions-constructs/aws-events-rule-kinesisstreams

aws-events-rule-lambda

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_events_rule_lambda`
TypeScript	`@aws-solutions-constructs/aws-events-rule-lambda`
Java	`software.amazon.awsconstructs.services.eventsrulelambda`

Overview

Este AWS Solutions Construct implementa uma regra de eventos da AWS e uma função do AWS Lambda.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
const { EventsRuleToLambdaProps, EventsRuleToLambda } from '@aws-solutions-constructs/
aws-events-rule-lambda';

const props: EventsRuleToLambdaProps = {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    },
    eventRuleProps: {
      schedule: events.Schedule.rate(Duration.minutes(5))
    }
};
```

```
new EventsRuleToLambda(this, 'test-events-rule-lambda', props);
```

Initializer

```
new EventsRuleToLambda(scope: Construct, id: string, props: EventsRuleToLambdaProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`EventsRuleToLambdaProps`

Adereços de criação de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo o tanto isso `elambdaFunctionProps` causará um erro.
LambdaFunctionProps	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.
EventruleProps	`events.RuleProps`	O usuário forneceu Eventrule Props para substituir os padrões

Propriedades de padrão

Nome	Tipo	Descrição
EventsRule	`events.Rule`	Retorna uma instância da regra Eventos criada pelo padrão.
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.

Configuração padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

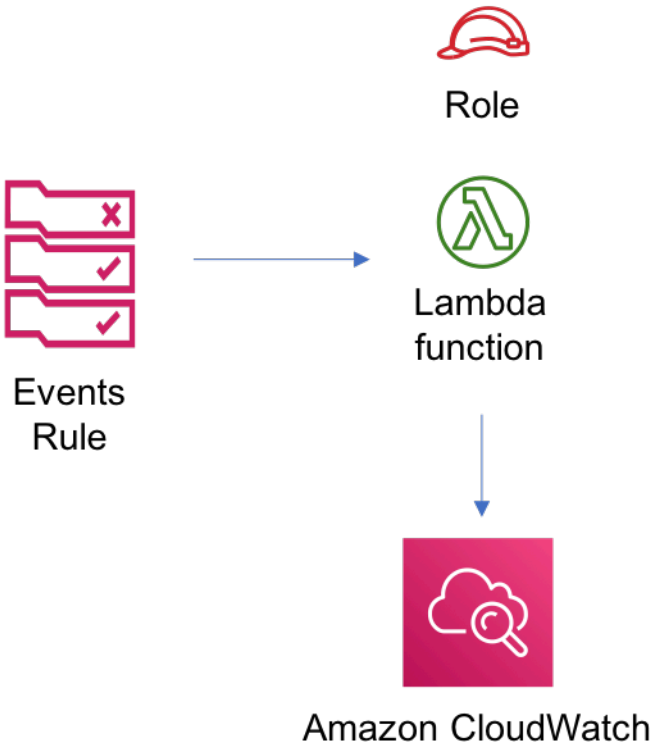
Amazon CloudWatch Events Regra

- Conceder permissões de menor privilégio ao CloudWatch Events para acionar a função Lambda

Função do AWS Lambda

- Configuração da função do IAM de acesso de privilégios limitados para Lambda
- Ativar a reutilização de conexões com a função Keep-Alive para NodeJS Lambda
- Ativar rastreamento do X-Ray
- Definição de variáveis de ambiente:
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

	@aws -solutions-constructs/aws-events-rule-lambda

aws-events-rule-sns

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_events_rule_sns`
TypeScript	`@aws-solutions-constructs/aws-events-rule-sns`
Java	`software.amazon.awsconstructs.services.eventsrulesns`

Overview

Esse padrão implementa uma regra do Amazon CloudWatch Events conectada a um tópico do Amazon SNS.

Aqui está uma definição de padrão implantável mínima:

```
import { Duration } from '@aws-cdk/core';
import * as events from '@aws-cdk/aws-events';
import * as iam from '@aws-cdk/aws-iam';
import { EventsRuleToSnsProps, EventsRuleToSns } from "@aws-solutions-constructs/aws-events-rule-sns";

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

const constructStack = new EventsRuleToSns(this, 'test-construct', props);

// 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);
```

Initializer

```
new EventsRuleToSNS(scope: Construct, id: string, props: EventsRuleToSNSProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`EventsRuleToSnsProps`

Adereços de construção de padrão

Nome	Tipo	Descrição
EventruleProps	`events.RuleProps`	Propriedades fornecidas pelo usuário para substituir as propriedades padrão da regra CloudWatch Events.
ExistingTopicobj?	`sns.Topic`	Instância existente do objeto Tópico SNS, fornecendo tanto isso quanto`topicProps` causará um erro.
TopicProps?	`sns.TopicProps`	Propriedades opcionais fornecidas pelo usuário para substituir as proprieda des padrão do tópico SNS. Ignorado se um`existingTopicObj` é fornecido.

Nome	Tipo	Descrição
EnableEncryptionWithCustomerManagedKey?	`boolean`	Se deve usar uma chave de criptografia gerenciada pelo cliente, gerenciada por este aplicativo CDK ou importada. Se importar uma chave de criptografia, ela deve ser especificada no campo `encryptionKey` propriedade para esta construção.
encryptionKey?	`kms.Key`	Uma chave de criptografia opcional existente a ser usada em vez da chave de criptografia padrão.
EncryptionKeyProps?	`kms.KeyProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da chave de criptografia.

Propriedades de padrão

Nome	Tipo	Descrição
EventsRule	`events.Rule`	Retorna uma instância da regra Eventos criada pelo padrão.
snsTopic	`sns.Topic`	Retorna uma instância do tópico SNS criado pelo padrão.

Nome	Tipo	Descrição
encryptionKey	`kms.Key`	Retorna uma instância da chave de criptografia criada pelo padrão.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

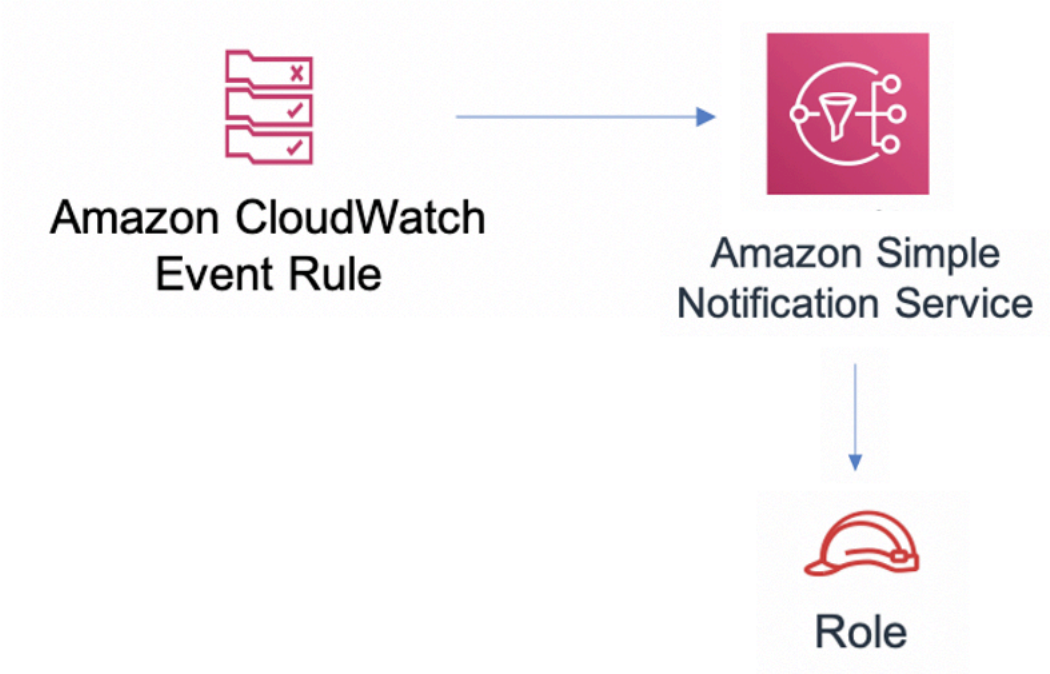
Regras de Amazon CloudWatch Events

- Conceda permissões de menor privilégio ao CloudWatch Events para publicar no tópico SNS.

Tópico do Amazon SNS

- Configurar permissões de acesso de menor privilégio para o tópico SNS.
- Habilitar a criptografia no lado do servidor para o tópico do SNS usando a chave do AWS KMS gerenciada pelo cliente.
- Aplique a criptografia de dados em trânsito

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-events-rule-sns

aws-events-rule-sqs

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àControle

de versão semânticaModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_events_rule_sqs`
TypeScript	`@aws-solutions-constructs/aws-events-rule-sqs`
Java	`software.amazon.awsconstructs.services.eventsrulesqs`

Overview

Esse padrão implementa uma regra do Amazon CloudWatch Events conectada a uma fila do Amazon SQS.

Aqui está uma definição de padrão implantável mínima:

```
import { Duration } from '@aws-cdk/core';
import * as events from '@aws-cdk/aws-events';
import * as iam from '@aws-cdk/aws-iam';
import { EventsRuleToSqsProps, EventsRuleToSqs } from "@aws-solutions-constructs/aws-events-rule-sqs";

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

const constructStack = new EventsRuleToSqs(this, 'test-construct', props);
```

```
// 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);
```

Initializer

```
new EventsRuleToSqs(scope: Construct, id: string, props: EventsRuleToSqsProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`EventsRuleToSqsProps`

Adereços de construção de padrão

Nome	Tipo	Descrição
EventruleProps	`events.RuleProps`	Propriedades fornecidas pelo usuário para substituir as propriedades padrão da regra CloudWatch Events.
ExistingQueueobj?	`sqs.Queue`	Uma fila SQS opcional existente a ser usada em vez da fila padrão. Fornecendo tanto isso quanto`queueProps` causará um erro.

Nome	Tipo	Descrição
QueueProps?	`sqs.QueueProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da fila SQS. Ignorado se um`existingQueueObj` é fornecido.
EnableQueuePurging?	`boolean`	Se deve conceder permissões adicionais para a função do Lambda, habilitando-o a limpar a fila SQS. Padronizado como `false`.
ImplantyDeadletterQueue?	`boolean`	Se criar uma fila secundária para ser usada como uma dead letter letter queue. Padronizado como `true`.
DeadletterQueueProps?	`sqs.QueueProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para a fila de letras inativas. Usado somente se o`deployDeadLetterQueue` está definida como true.
MaxReceiveCount?	`number`	O número de vezes que uma mensagem pode ser desenfileirada sem sucesso antes de ser movida para a fila de mensagens mortas. Padronizado como 15.

Nome	Tipo	Descrição
EnableEncryptionWithCustomerManagedKey?	`boolean`	Se deve usar uma chave de criptografia gerenciada pelo cliente, gerenciada por este aplicativo CDK ou importada. Se importar uma chave de criptografia, ela deve ser especificada na caixa `encryptionKey` propriedade para esta construção.
encryptionKey?	`kms.Key`	Uma chave de criptografia opcional existente a ser usada em vez da chave de criptografia padrão.
EncryptionKeyProps?	`kms.KeyProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da chave de criptografia.

Propriedades de padrão

Nome	Tipo	Descrição
EventsRule	`events.Rule`	Retorna uma instância da regra Eventos criada pelo padrão.
SQSqueue	`sqs.Queue`	Retorna uma instância da fila SQS criada pelo padrão.

Nome	Tipo	Descrição
encryptionKey	`kms.Key`	Retorna uma instância da chave de criptografia criada pelo padrão.
DeadletterQueue?	`sqs.Queue`	Retorna uma instância da fila de letras mortas criada pelo padrão, se uma for implantad a.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

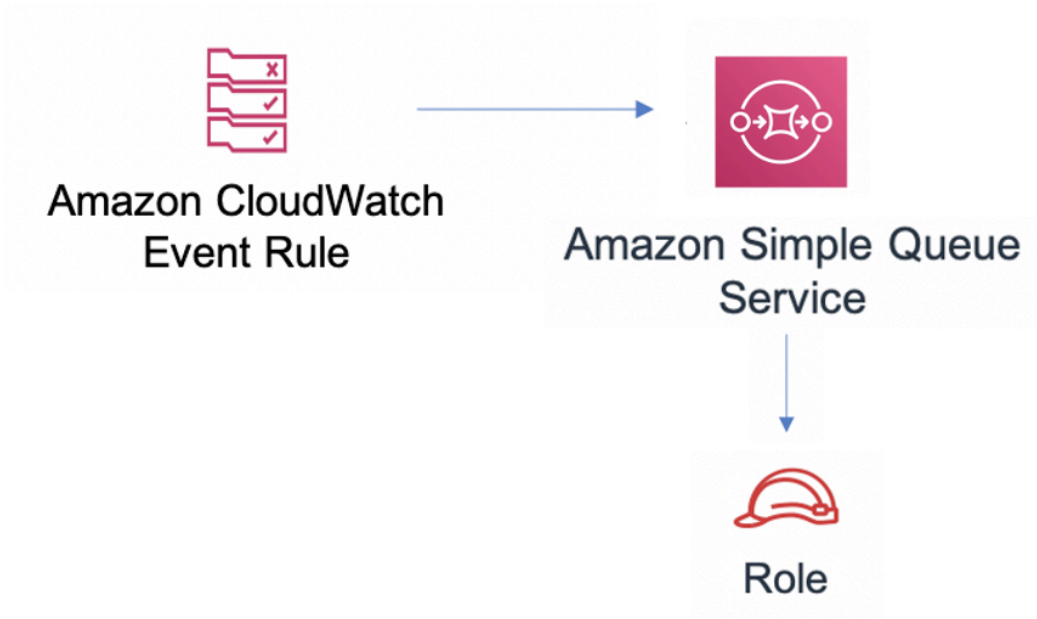
Regras de Amazon CloudWatch Events

- Conceda permissões de menor privilégio ao CloudWatch Events para publicar na Fila SQS.

Fila do Amazon SQS

- Implante uma fila de mensagens mortas para a fila de origem.
- Ative a criptografia do lado do servidor para a fila de origem usando uma chave do AWS KMS gerenciada pelo cliente.
- Aplique a criptografia de dados em trânsito

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-events-rule-sqs

aws-events-rule-step-function

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àControle

de versão semânticaModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_events_rule_step_function`
TypeScript	`@aws-solutions-constructs/aws-events-rule-step-function`
Java	`software.amazon.awsconstructs.services.eventsrulestepfunction`

Overview

Este AWS Solutions Construct implementa uma regra de eventos da AWS e uma função AWS Step.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { EventsRuleToStepFunction, EventsRuleToStepFunctionProps } from '@aws-solutions-constructs/aws-events-rule-step-function';

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

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

```
new EventsRuleToStepFunction(this, 'test-events-rule-step-function-stack', props);
```

Initializer

```
new EventsRuleToStepFunction(scope: Construct, id: string, props:
 EventsRuleToStepFunctionProps);
```

Parâmetros

- escopoConstruct
- idstring
- propsEventsRuleToStepFunctionProps

Adereços de criação de padrão

Nome	Tipo	Descrição
StateMachineProps	sfn.StateMachineProps	O usuário opcional forneceu adereços para substituir os adereços padrão para SFN.StateMachine
EventruleProps	events.RuleProps	O usuário forneceu Eventrule Props para substituir os padrões
CreateCloudWatchAlms	boolean	Criar alarmes recomendados do CloudWatch.
LoggroupProps?	logs.LogGroupProps	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o grupo de logs do CloudWatch Logs.

Propriedades de padrão

Nome	Tipo	Descrição
CloudwatchAlarm?	`cloudwatch.Alarm[]`	Retorna uma lista de um ou mais alarmes do CloudWatch criados pelo padrão.
EventsRule	`events.Rule`	Retorna uma instância da regra Eventos criada pelo padrão.
StateMachine	`sfn.StateMachine`	Retorna uma instância da máquina de estado criada pelo padrão.
StateMachineLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para a máquina de estado.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

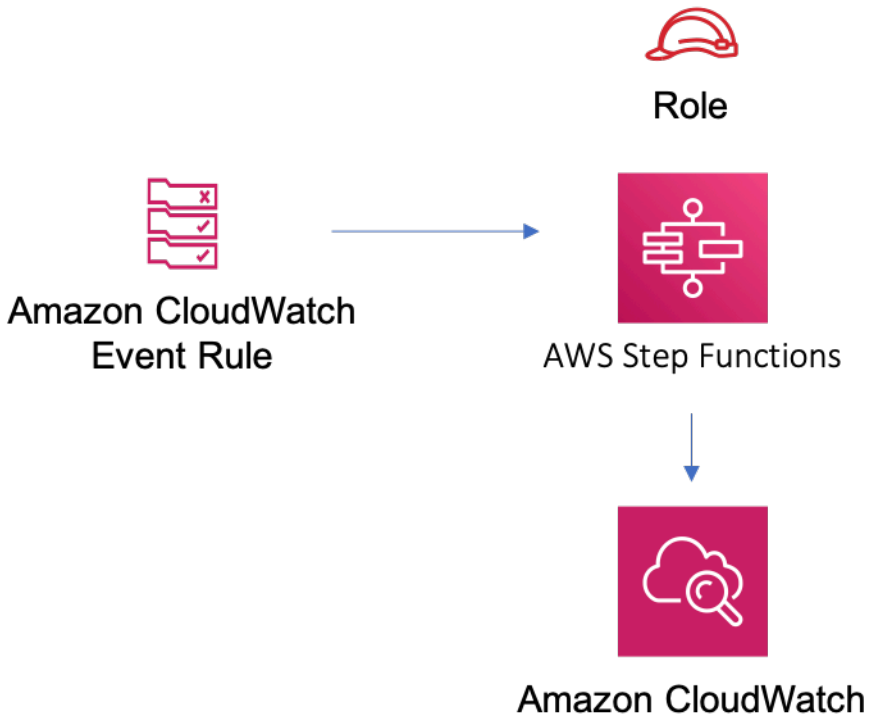
Amazon CloudWatch Events

- Conceder permissões de menor privilégio ao CloudWatch Events para acionar a função do Lambda

AWS Step Function

- Habilitar o log do CloudWatch para o
- Implante os alarmes do CloudWatch de práticas recomendadas para a função Step

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-events-rule-step-function

aws-iot-kinesisfirehose-s3

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_iot_kinesisfirehose_s3`
TypeScript	`@aws-solutions-constructs/aws-iot-kinesisfirehose-s3`
Java	`software.amazon.awsconstructs.services.iotkinesisfirehoses3`

Overview

Este AWS Solutions Construct implementa uma regra de tópico do AWS IoT MQTT para enviar dados para um stream de entrega do Amazon Kinesis Data Firehose conectado a um bucket do Amazon S3.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { IotToKinesisFirehoseToS3Props, IotToKinesisFirehoseToS3 } from '@aws-solutions-constructs/aws-iot-kinesisfirehose-s3';

const props: 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', props);
```

Initializer

```
new IotToKinesisFirehoseToS3(scope: Construct, id: string, props:
 IotToKinesisFirehoseToS3Props);
```

Parâmetros

- escopoConstruct
- idstring
- propsIotToKinesisFirehoseToS3Props

Adereços de criação de padrão

Nome	Tipo	Descrição
IOtTopicRuleProps	iot.CfnTopicRuleProps	O usuário forneceu CFNTopicRuleProps para substituir os padrões
Kinesis é Firehoseprops?	kinesisfirehose.CfnDeliveryStreamProps	Os adereços fornecidos pelo usuário opcionais para substituir os adereços padrão do Kinesis Firehose Delivery Stream
ExistingBucketoBJ?	s3.Bucket	Instância existente do objeto S3 Bucket, fornecendo isso ebucketProps causará um erro.
Baldes?	s3.BucketProps	O usuário forneceu adereços para substituir os adereços padrão para o bucket do S3. Se isso for fornecido,

Nome	Tipo	Descrição
		então também fornecend `obucketProps` é um erro.
LoggroupProps?	`logs.LogGroupProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o grupo de logs do CloudWatch Logs.

Propriedades do padrão

Nome	Tipo	Descrição
IOtActionsRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão para a regra de IoT.
IOtTopicRule	`iot.CfnTopicRule`	Retorna uma instância da regra de tópico IoT criada pelo padrão.
KinesisFirehose	`kinesisfirehose.CfnDeliveryStream`	Retorna uma instância do stream de entrega do Kinesis Firehose criado pelo padrão.
KinesisFireHoseLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para o qual os logs de acesso do Kinesis Firehose são enviados.
KinesisFireHoserole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão para o stream de entrega do Kinesis Firehose.

Nome	Tipo	Descrição
S3 Bucket?	`s3.Bucket`	Retorna uma instância do bucket S3 criado pelo padrão.
S3loggingBucket?	`s3.Bucket`	Retorna uma instância do bucket de log criado pelo padrão para o bucket S3.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon IoT

- Configurar a função do IAM de acesso de menor privilégio para o Amazon IoT

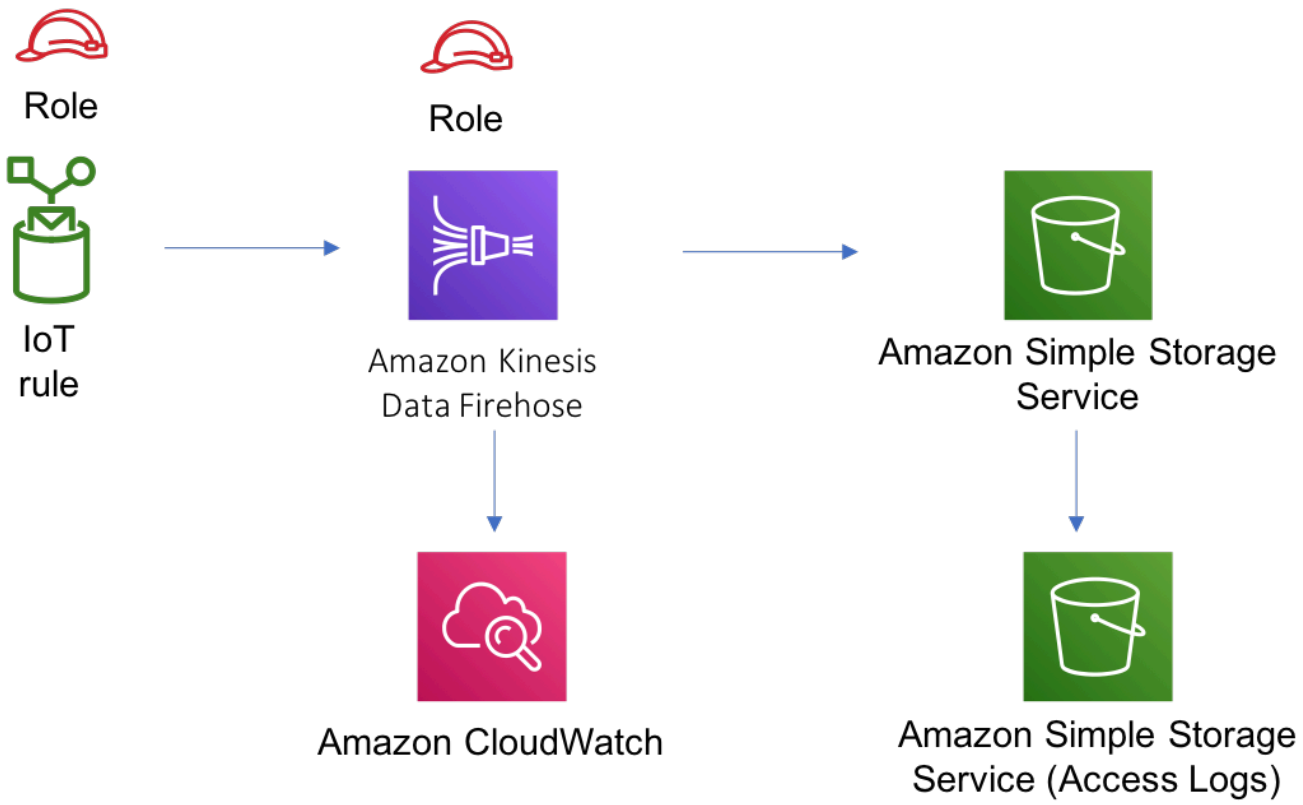
Amazon Kinesis Firehose

- Ativar o registro do CloudWatch para o Kinesis Firehose
- Configurar a função do IAM de acesso de menor privilégio para o Amazon Kinesis Firehose

Bucket do Amazon S3

- Configurar registro de acesso para o bucket do S3
- Ativar criptografia no lado do servidor para o S3 Bucket usando a chave KMS gerenciada pela AWS
- Ativar o controle de versão para o bucket do S3
- Não permitir acesso público para o S3 Bucket
- Manter o bucket do S3 ao excluir a pilha do CloudFormation
- Aplica regra de ciclo de vida para mover versões de objetos não atuais para o armazenamento do Glacier após 90 dias

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-iot-kinesisfirehose-s3

aws-iot-lambda

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_iot_lambda`
TypeScript	`@aws-solutions-constructs/aws-iot-lambda`
Java	`software.amazon.awsconstructs.services.iotlambda`

Overview

Este padrão de constrói soluções da AWS implementa uma regra de tópico do AWS IoT MQTT e um padrão de função do AWS Lambda.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { IotToLambdaProps, IotToLambda } from '@aws-solutions-constructs/aws-iot-
lambda';

const props: IotToLambdaProps = {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        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', props);
```

Initializer

```
new IotToLambda(scope: Construct, id: string, props: IotToLambdaProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`IotToLambdaProps`

Adereços de construção de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo tanto isso `elambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.

Nome	Tipo	Descrição
IotTopicRuleProps?	`iot.CfnTopicRulePr ops`	O usuário forneceu CFNTopicRuleProps para substituir os padrões

Propriedades de padrão

Nome	Tipo	Descrição
IOtTopicRule	`iot.CfnTopicRule`	Retorna uma instância da regra de tópico IoT criada pelo padrão.
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

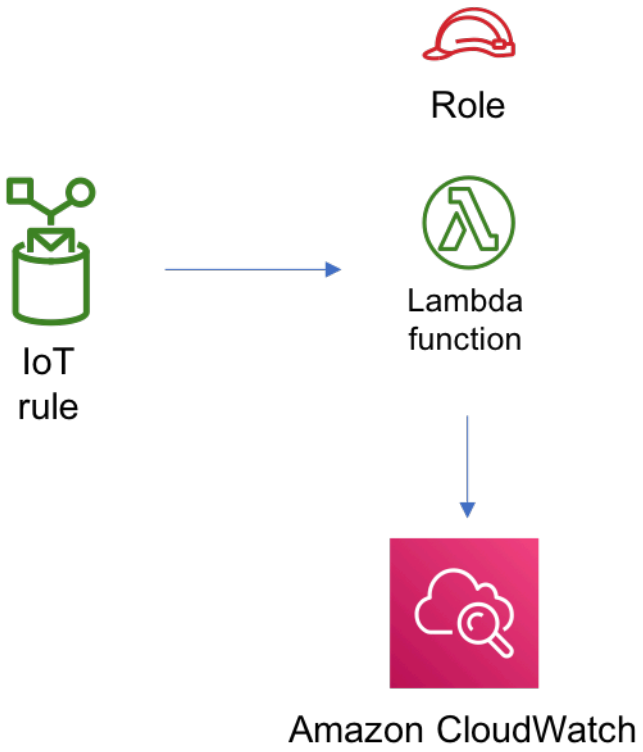
Regra do Amazon IoT

- Configurar a função do IAM de acesso de menor privilégio para o Amazon IoT.

Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Ativar rastreamento de X-Ray.
- SET variáveis de ambiente:
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-iot-lambda

aws-iot-lambda-dynamodb

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_iot_lambda_dynamodb`
TypeScript	`@aws-solutions-constructs/aws-iot-lambda-dynamodb`
Java	`software.amazon.awsconstructs.services.iotlambdadynamodb`

Overview

Este padrão de constrói soluções da AWS implementa uma regra de tópico do AWS IoT, uma função do AWS Lambda e uma tabela do Amazon DynamoDB com as permissões menos privilegiadas.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { IotToLambdaToDynamoDBProps,  IotToLambdaToDynamoDB } from '@aws-solutions-
constructs/aws-iot-lambda-dynamodb';

const props: IotToLambdaToDynamoDBProps = {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        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', props);
```

Initializer

```
new IotToLambdaToDynamoDB(scope: Construct, id: string, props:
 IotToLambdaToDynamoDBProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`IotToLambdaToDynamoDBProps`

Props de criação de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo o tanto isso `elambdaFunctionProps` causará um erro.
LambdaFunctionProps	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.
IOtTopicRuleProps	`iot.CfnTopicRuleProps`	O usuário forneceu adereços para substituir os adereços padrão

Nome	Tipo	Descrição
DynamotableProps?	`dynamodb.TableProps`	Os adereços fornecidos pelo usuário opcionais para substituir os adereços padrão da Tabela do DynamoDB
TablePermissions?	`string`	Permissões de tabela opcionais a serem concedida s à função do Lambda. Uma das seguintes opções podem ser especific adas:`All`,`Read`,`ReadWrite` , ou`Write`.

Propriedades de padrão

Nome	Tipo	Descrição
DynamoTable	`dynamodb.Table`	Retorna uma instância da tabela do DynamoDB criada pelo padrão.
IOtTopicRule	`iot.CfnTopicRule`	Retorna uma instância da regra de tópico IoT criada pelo padrão.
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon IoT Regra

- Configurar a função do IAM de acesso de menor privilégio para o Amazon IoT.

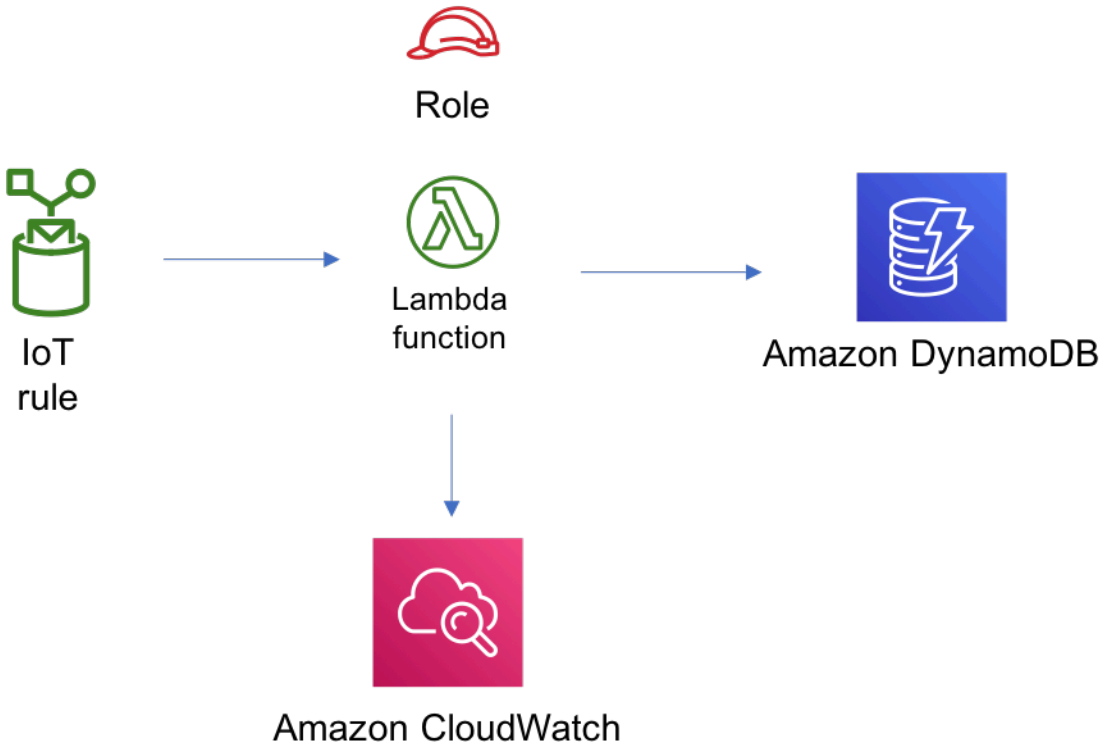
Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Ativar rastreamento do X-Ray.
- Definir variáveis de ambiente:
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Amazon DynamoDB Tabela

- Defina o modo de faturamento da Tabela do DynamoDB como On-Demand (Pagamento por solicitação).
- Habilite a criptografia do lado do servidor para a tabela do DynamoDB usando a chave KMS gerenciada pela AWS.
- Cria uma chave de partição chamada 'id' para a tabela do DynamoDB.
- Manter a tabela ao excluir a pilha do CloudFormation.
- Permita backups contínuos e recuperação point-in-time.

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

	@aws -solutions-constructs/aws-iot-lambda-dynamodb

aws-kinesisfirehose-s3

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àControle de versão semânticaModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws-kinesis-firehose-s3`
TypeScript	`@aws-solutions-constructs/aws-kinesisfirehose-s3`
Java	`software.amazon.awsconstructs.services.kinesisfirehoses3`

Overview

Este AWS Solutions Construct implementa um fluxo de entrega do Amazon Kinesis Data Firehose conectado a um bucket do Amazon S3.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { KinesisFirehoseToS3 } from '@aws-solutions-constructs/aws-kinesisfirehose-s3';

new KinesisFirehoseToS3(this, 'test-firehose-s3', {});
```

Initializer

```
new KinesisFirehoseToS3(scope: Construct, id: string, props: KinesisFirehoseToS3Props);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`KinesisFirehoseToS3Props`

Props de criação de padrão

Nome	Tipo	Descrição
Baldes?	`s3.BucketProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para o bucket do S3.
ExistingBucketoBJ?	`s3.IBucket`	Instância existente opcional do S3 Bucket. Se isso for fornecido, então também fornecendo`bucketProps` é um erro.
ExistingLoggingBucketoBJ?	`s3.IBucket`	Instância existente opcional de log do S3 Bucket para o S3 Bucket criado pelo padrão.
Kinesis é Firehoseprops?	`kinesisfirehose.CfnDeliveryStreamProps` | `any`	O usuário opcional forneceu adereços para substituir os adereços padrão do Kinesis Firehose Delivery Stream.
LoggroupProps?	`logs.LogGroupProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para o CloudWatchLogs LogGroup.

Propriedades do padrão

Nome	Tipo	Descrição
KinesisFirehose	`kinesisfirehose.CfnDeliveryStream`	Retorna uma instância de KinesisFireHose.cfnDelivery

Nome	Tipo	Descrição
		Stream criada pela construçã o.
KinesisFireHoseLogGroup	`logs.LogGroup`	Retorna uma instância do logs.logGroup criado pela construção para o stream de entrega do Kinesis Data Firehose.
KinesisFireHoserole	`iam.Role`	Retorna uma instância do IAM.Role criado pela construção para o stream de entrega do Kinesis Data Firehose.
S3 Bucket?	`s3.Bucket`	Retorna uma instância de S3.bucket criada pela construção.
S3loggingBucket?	`s3.Bucket`	Retorna uma instância de S3.bucket criada pela construção como o bucket de log para o bucket primário.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon Kinesis Firehose

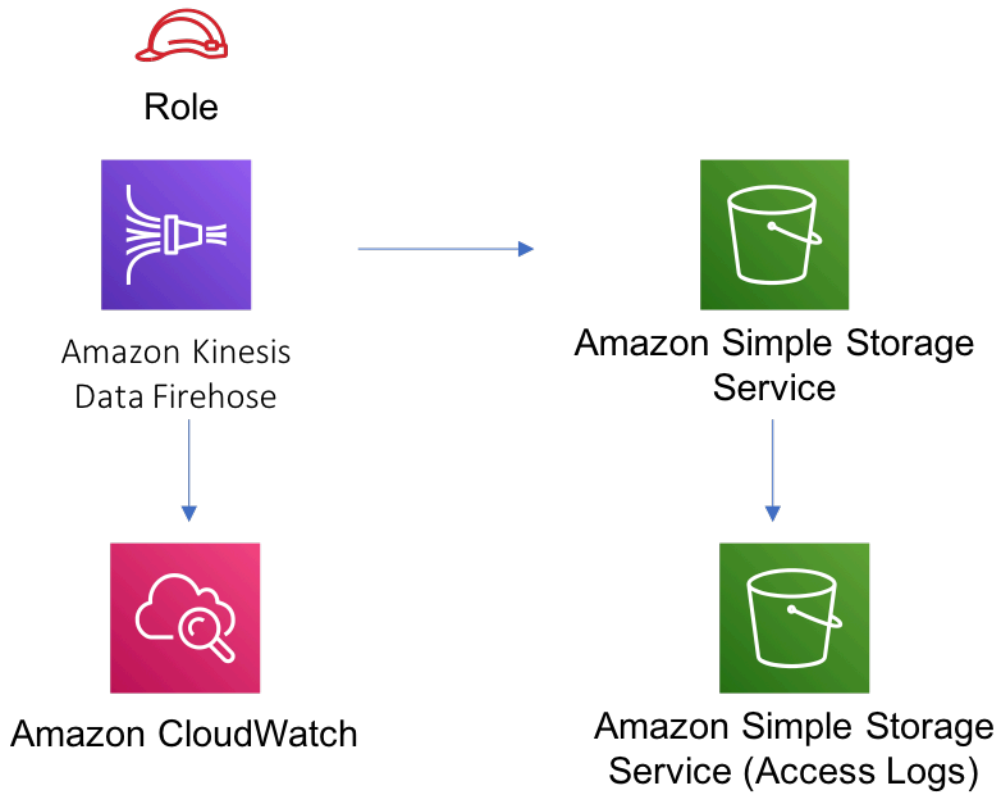
- Ativar o registro do CloudWatch para o Kinesis Firehose
- Configurar a função do IAM de acesso de menor privilégio para o Amazon Kinesis Firehose

Amazon S3 Bucket

- Configurar registro de acesso para o bucket do S3

- Ativar criptografia no lado do servidor para o S3 Bucket usando a chave KMS gerenciada pela AWS
- Ativar o controle de versão para o bucket do S3
- Não permitir acesso público para o S3 Bucket
- Manter o bucket do S3 ao excluir a pilha do CloudFormation
- Aplique a criptografia de dados em trânsito
- Aplica regra de ciclo de vida para mover versões de objetos não atuais para o armazenamento do Glacier após 90 dias

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

	@aws -solutions-constructs/aws-kinesisfirehose-s3

aws-kinesisfirehose-s3-e-kinesisanalytics

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àSemantic VersioningModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_kinesisfirehose_s3_and_kinesisanalytics`
TypeScript	`@aws-solutions-constructs/aws-kinesisfirehose-s3-and-kinesisanalytics`
Java	`software.amazon.awsconstructs.services.kinesisfirehose s3kinesisanalytics`

Overview

Este AWS Solutions Construct implementa um stream de entrega do Amazon Kinesis Firehose conectado a um bucket do Amazon S3 e a um aplicativo do Amazon Kinesis Analytics.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { KinesisFirehoseToAnalyticsAndS3 } from '@aws-solutions-constructs/aws-
kinesisfirehose-s3-and-kinesisanalytics';

new KinesisFirehoseToAnalyticsAndS3(this, 'FirehoseToS3AndAnalyticsPattern', {
    kinesisAnalyticsProps: {
```

```
        inputs: [{
            inputSchema: {
                recordColumns: [{
                    name: 'ticker_symbol',
                    sqlType: 'VARCHAR(4)',
                    mapping: '$.ticker_symbol'
                }, {
                    name: 'sector',
                    sqlType: 'VARCHAR(16)',
                    mapping: '$.sector'
                }, {
                    name: 'change',
                    sqlType: 'REAL',
                    mapping: '$.change'
                }, {
                    name: 'price',
                    sqlType: 'REAL',
                    mapping: '$.price'
                }],
                recordFormat: {
                    recordFormatType: 'JSON'
                },
                recordEncoding: 'UTF-8'
            },
            namePrefix: 'SOURCE_SQL_STREAM'
        }]
    }
});
```

Initializer

```
new KinesisFirehoseToAnalyticsAndS3(scope: Construct, id: string, props:
 KinesisFirehoseToAnalyticsAndS3Props);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`KinesisFirehoseToAnalyticsAndS3Props`

Padrão de criação

Nome	Tipo	Descrição
Kinesis é Firehoseprops?	`kinesisFirehose.CfnDeliveryStreamProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão do stream de entrega do Kinesis Firehose.
KinesisAnalyticsProps?	`kinesisAnalytics.CfnApplicationProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão do aplicativo Kinesis Analytics.
ExistingBucketoBJ?	`s3.IBucket`	Instância existente do objeto S3 Bucket. Se isso for fornecido, então também fornecendo`bucketProps` é um erro. |
Baldes?	`s3.BucketProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão do bucket. Ignorado se um`existingBucketObj` é fornecido.
LoggroupProps?	`logs.LogGroupProps`	Opcionais fornecidos pelo usuário para substituir os adereços padrão para o grupo de logs do CloudWatch Logs.

Propriedades de padrão

Nome	Tipo	Descrição
KinesisAnalytics	`kinesisAnalytics.CfnApplication`	Retorna uma instância do aplicativo Kinesis Analytics criado pelo padrão.
KinesisFirehose	`kinesisfirehose.CfnDeliveryStream`	Retorna uma instância do stream de entrega do Kinesis Firehose criado pelo padrão.
KinesisFireHoseLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para o qual os logs de acesso do Kinesis Firehose são enviados.
KinesisFireHoserole	`iam.Role`	Retorna uma instância da função do IAM criada pelo padrão para o stream de entrega do Kinesis Firehose.
S3Bucket?	`s3.Bucket`	Retorna uma instância do bucket S3 criado pelo padrão.
S3loggingBucket?	`s3.Bucket`	Retorna uma instância do bucket de log criado pelo padrão para o bucket S3.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon Kinesis Firehose

- Ativar o registro do CloudWatch para o Kinesis Firehose
- Configurar a função do IAM de acesso de menor privilégio para o Amazon Kinesis Firehose

Amazon S3 Bucket

- Configurar registro de acesso para o bucket do S3
- Ativar criptografia no lado do servidor para o S3 Bucket usando a chave KMS gerenciada pela AWS
- Ativar o controle de versão para o bucket do S3
- Não permitir acesso público para o S3 Bucket
- Manter o bucket do S3 ao excluir a pilha do CloudFormation
- Aplique a criptografia de dados em trânsito
- Aplica regra de ciclo de vida para mover versões de objetos não atuais para o armazenamento do Glacier após 90 dias

Amazon Kinesis Data Analytics

- Configurar a função do IAM de acesso de menor privilégio para o Amazon Kinesis Analytics

Architecture

GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-kinesisfirehose-s3-and-kinesisanalytics

aws-kinesisstreams-gluejob

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_kinesis_streams_gluejob`
TypeScript	`@aws-solutions-constructs/aws-kinesisstreams-gluejob`
Java	`software.amazon.awsconstructs.services.kinesisstreamsgluejob`

Overview

Este AWS Solutions Construct implanta um Amazon Kinesis Data Stream e configura um AWS Glue Job para executar a transformação ETL personalizada com os recursos/propriedades apropriados

para interação e segurança. Ele também cria um bucket do Amazon S3 no qual o script Python para o AWS Glue Job pode ser carregado.

Aqui está uma definição de padrão implantável mínima no 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',
            scriptLocation: new s3assets.Asset(this, 'ScriptLocation', {
                path: `${__dirname}/../etl/transform.py`,
            }).s3ObjectUrl,
        },
    },
```

```
    fieldSchema: fieldSchema,
});
```

Initializer

```
new KinesisstreamsToGluejob(scope: Construct, id: string, props:
 KinesisstreamsToGluejobProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`KinesisstreamsToGluejobProps`

Adereços de construção padrão

Nome	Tipo	Descrição
KinesisStreamprops?	`kinesis.StreamProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o fluxo de dados do Amazon Kinesis.
ExistingStreamobj?	`kinesis.Stream`	Instância existente do Kinesis Stream, fornecendo tanto isso quanto`kinesisStreamProps` causará um erro.
GlueJobprops?	`cfnJob.CfnJobProps`	Props fornecidos pelo usuário para substituir os adereços padrão para o trabalho do AWS Glue.

Nome	Tipo	Descrição
ExistingGlueJob?	`cfnJob.CfnJob`	Instância existente do AWS Glue Job, fornecendo tanto isso quanto`glueJobProps` causará um erro.
ExistingDatabase?	`CfnDatabase`	Banco de dados existente do AWS Glue para ser usado com essa construção. Se isso estiver definido, então`databaseProps` É ignorado.
DatabaseProps?	`CfnDatabaseProps`	Props fornecidos pelo usuário para substituir os adereços padrão usados para criar o banco de dados do AWS Glue.
ExistingTable?	`CfnTable`	Instância existente da tabela AWS Glue. Se isso estiver definido, então`tableProps` e`fieldSchema` são ignorados.
Maçonetes?	`CfnTableProps`	Props fornecidos pelo usuário para substituir adereços padrão usados para criar uma tabela do AWS Glue.
FieldSchema?	`CfnTable.ColumnProperty[]`	Estrutura de esquema fornecida pelo usuário para criar uma tabela do AWS Glue.

Nome	Tipo	Descrição
Saída DataStore?	`SinkDataStoreProps`	Props fornecidos pelo usuário para um bucket do Amazon S3 que armazena a saída do trabalho do AWS Glue. Atualmente só é compatível com o Amazon S3 como o tipo de armazenamento de dados de saída.

SinkDataStoreProps

Nome	Tipo	Descrição
Existings3OutputBucket?	`Bucket`	Instância existente do bucket do S3 em que os dados devem ser gravados. Fornecendo tanto isso quanto`outputBucketProps` causará um erro.
OutputBucketProps	`BucketProps`	Propriedades de bucket fornecidas pelo usuário para criar o bucket do Amazon S3 usado para armazenar a saída do trabalho do AWS Glue.
DataStoreType	`SinkStoreType`	Tipo de armazenamento de dados do coletor

SinkStoreType

Enumeração de tipos de armazenamento de dados que podem incluir S3, DynamoDB, DocumentDB, RDS ou Redshift. Implementação de construção atual suporta apenas S3, mas potencial para adicionar outros tipos de saída no futuro.

Nome	Tipo	Descrição
S3	`string`	Tipo de armazenamento do S3

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon Kinesis Stream

- Configure a função do IAM de acesso de menor privilégio para o fluxo de dados do Amazon Kinesis.
- Ative a criptografia no lado do servidor para o Amazon Kinesis Stream usando uma chave KMS gerenciada da AWS.
- Implante os alarmes de práticas recomendadas do Amazon CloudWatch para o Amazon Kinesis Stream.

Job Glue

- Crie uma configuração de segurança do AWS Glue que configura a criptografia para CloudWatch, Job Bookmarks e S3. CloudWatch e Job Bookmarks são criptografados usando a AWS Managed KMS Key criada para o AWS Glue Service. O bucket S3 é configurado com o modo de criptografia SSE-S3.
- Configurar políticas de função de serviço que permitem que o AWS Glue leia do Amazon Kinesis Data Streams.

Banco de dados cola

- Crie um banco de dados AWS Glue. Uma tabela do AWS Glue será adicionada ao banco de dados. Esta tabela define o esquema para os registros armazenados em buffer no Amazon Kinesis Data Stream.

Mesa Glue

- Crie uma tabela do AWS Glue. A definição do esquema de tabela é baseada na estrutura JSON dos registros armazenados em buffer no Amazon Kinesis Data Stream.

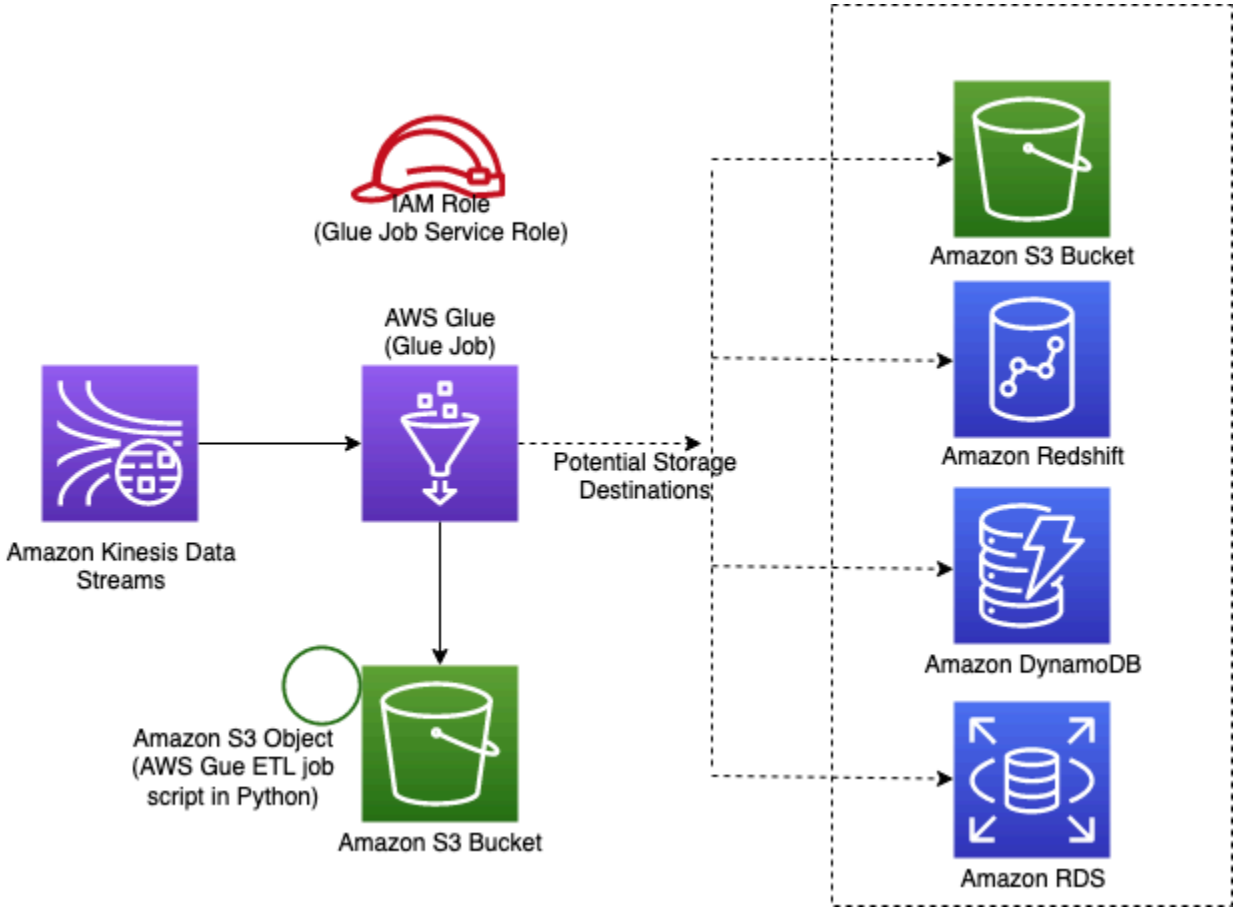
Função do IAM

- Uma função de execução de trabalho que tem privilégios para 1) ler o script ETL no local do bucket do Amazon S3, 2) ler registros do Amazon Kinesis Data Stream e 3) executar o trabalho do Amazon Glue.

Balde S3 de saída

- Um bucket do Amazon S3 para armazenar a saída da transformação do ETL. Esse bucket será passado como um argumento para o trabalho criado do AWS Glue para que ele possa ser usado no script ETL para gravar dados nele.

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

	@aws -solutions-constructs/aws-kinesisstreams-gluejob

aws-kinesisstreams-kinesisfirehose-s3

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
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

Este AWS Solutions Construct implementa um Amazon Kinesis Data Stream (KDS) conectado ao stream de entrega do Amazon Kinesis Data Firehose (KDF) conectado a um bucket do Amazon S3.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { KinesisStreamsToKinesisFirehoseToS3 } from '@aws-solutions-constructs/aws-
kinesisstreams-kinesisfirehose-s3';

new KinesisStreamsToKinesisFirehoseToS3(this, 'test-stream-firehose-s3', {});
```

Initializer

```
new KinesisStreamsToKinesisFirehoseToS3(scope: Construct, id: string, props:
 KinesisStreams...ToS3Props);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`KinesisStreams...ToS3Props`

Adereços de criação de padrão

Nome	Tipo	Descrição
Baldes?	`s3.BucketProps`	Os adereços fornecidos pelo usuário opcionais para substituir os adereços padrão para o bucket do S3.
CreateCloudWatchalarms?	`boolean`	Opcional se deseja criar alarmes recomendados do CloudWatch.
ExistingBucketoBJ?	`s3.IBucket`	Instância existente opcional do objeto S3 Bucket. Se isso for fornecido, então também fornecendo`bucketProps` é um erro.
ExistingLoggingBucketoBJ?	`s3.IBucket`	Instância existente opcional do objeto S3 Bucket de log para o S3 Bucket criado pelo padrão.
ExistingStreamobj?	`kinesis.Stream`	Instância existente do Kinesis Stream, fornecendo tanto isso quanto`kinesisStreamProps` causará um erro.
Kinesis é Firehoseprops?	`aws-kinesisfirehose.CfnDeliveryStreamProps` | `any`	O usuário opcional forneceu adereços para substituir os adereços padrão do Kinesis Firehose Delivery Stream.

Nome	Tipo	Descrição
KinesisStreamprops?	`kinesis.StreamProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para o stream do Kinesis.
LoggroupProps?	`logs.LogGroupProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para o Grupo de Logs do CloudWatchLogs.

Propriedades do padrão

Nome	Tipo	Descrição
CloudwatchAlarm?	`cloudwatch.Alarm[]`	Retorna uma lista de instância s do CloudWatch.Alarm criadas pela construção.
KinesisFirehose	`kinesisfirehose.CfnDeliveryStream`	Retorna uma instância de KinesisFireHose.cfnDelivery Stream criada pela construçã o.
KinesisFireHoseLogGroup	`logs.LogGroup`	Retorna uma instância do logs.logGroup criado pela construção para o stream de entrega do Kinesis Data Firehose.
KinesisFireHoserole	`iam.Role`	Retorna uma instância do IAM.Role criado pela construção para o stream de entrega do Kinesis Data Firehose.

Nome	Tipo	Descrição
KinesisStreamRole	`iam.Role`	Retorna uma instância do IAM.Role criado pela construção para o stream do Kinesis.
S3Bucket?	`s3.Bucket`	Retorna uma instância de S3.bucket criada pela construção.
S3loggingBucket?	`s3.Bucket`	Retorna uma instância de S3.bucket criada pela construção como o bucket de log para o bucket primário.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon Kinesis Stream

- Configurar a função do IAM de acesso de menor privilégio para Kinesis Stream
- Ativar criptografia no lado do servidor para o Kinesis Stream usando a chave KMS gerenciada da AWS
- Implantar práticas recomendadas Alarmes do CloudWatch para o Kinesis Stream

Amazon Kinesis Firehose

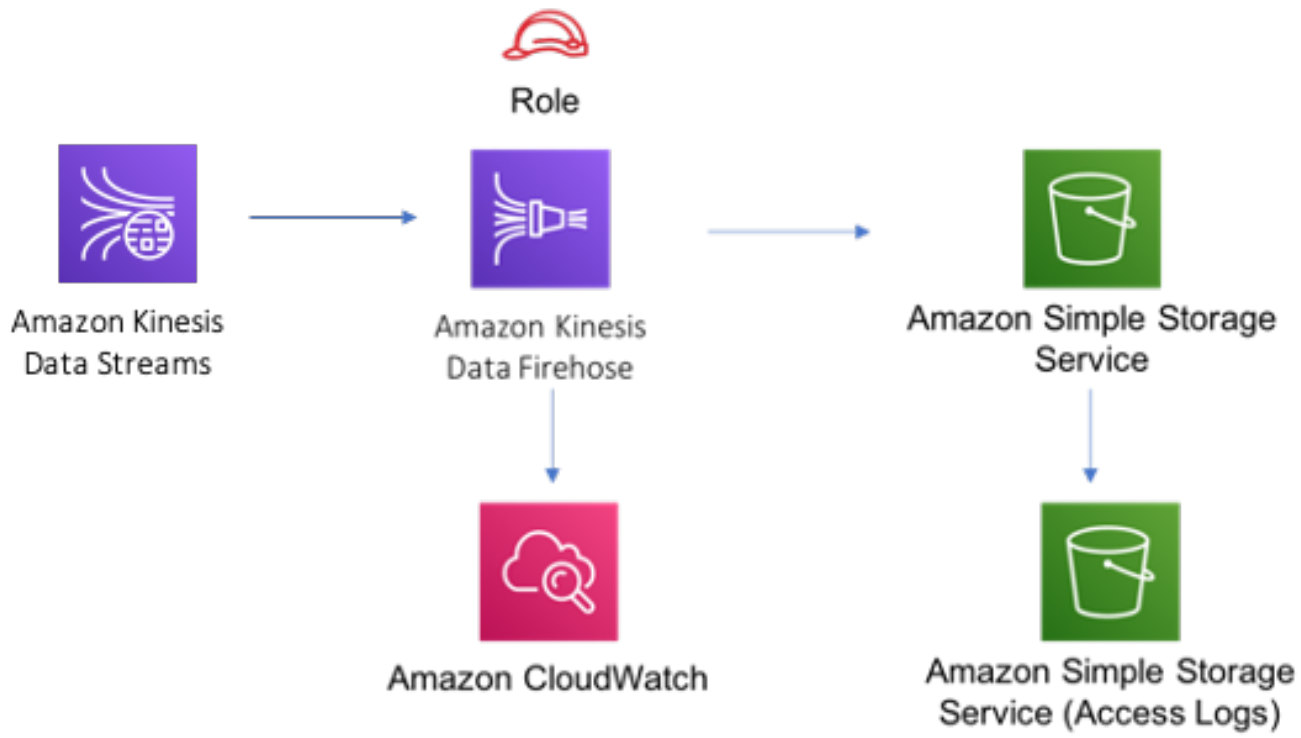
- Ativar o registro do CloudWatch para o Kinesis Firehose
- Configurar a função do IAM de acesso de menor privilégio para o Amazon Kinesis Firehose

Bucket do Amazon S3

- Configurar registro de acesso para bucket do S3
- Ativar criptografia no lado do servidor para bucket do S3 usando a chave KMS gerenciada pela AWS

- Aplique a criptografia de dados em trânsito
- Ativar controle de versão do bucket
- Não permitir acesso público para bucket do S3
- Manter o bucket do S3 ao excluir a pilha do CloudFormation
- Aplicar regra de ciclo de vida para mover versões de objetos não atuais para o armazenamento do Glacier após 90 dias

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

	[@aws -solutions-constructs/aws-kinesisstreams-kinesisfirehose-s3](#)

aws-kinesisstreams-lambda

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws-kinesis-streams-lambda`
TypeScript	`@aws-solutions-constructs/aws-kinesisstreams-lambda`
Java	`software.amazon.awsconstructs.services.kinesisstreamslambda`

Overview

Este AWS Solutions Construct implanta uma função Kinesis Stream e Lambda com os recursos/propriedades apropriados para interação e segurança.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { KinesisStreamsToLambda } from '@aws-solutions-constructs/aws-kinesisstreams-
lambda';

new KinesisStreamsToLambda(this, 'KinesisToLambdaPattern', {
    kinesisEventSourceProps: {
        startingPosition: lambda.StartingPosition.TRIM_HORIZON,
        batchSize: 1
    },
    lambdaFunctionProps: {
```

```
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    }
});
```

Initializer

```
new KinesisStreamsToLambda(scope: Construct, id: string, props:
 KinesisStreamsToLambdaProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`KinesisStreamsToLambdaProps`

Props de criação de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo tanto isso `elambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.

Nome	Tipo	Descrição
KinesisStreamprops?	`kinesis.StreamProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão do stream do Kinesis.
ExistingStreamobj?	`kinesis.Stream`	Instância existente do Kinesis Stream, fornecendo tanto isso quanto`kinesisStreamProps` causará um erro.
KinesiSeventSourceProps?	`aws-lambda-event-sources.KinesisEventSourceProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o mapeamento de origem de evento do Lambda.
CreateCloudWatchAlms	`boolean`	Criar alarmes recomendados do CloudWatch.

Propriedades de padrão

Nome	Tipo	Descrição
KinesisStream	`kinesis.Stream`	Retorna uma instância do stream do Kinesis criado pelo padrão.
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.
KinesisStreamRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo

Nome	Tipo	Descrição
		padrão para o stream do Kinesis.
CloudwatchAlarm?	`cloudwatch.Alarm[]`	Retorna uma lista de um ou mais alarmes do CloudWatch criados pelo padrão.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

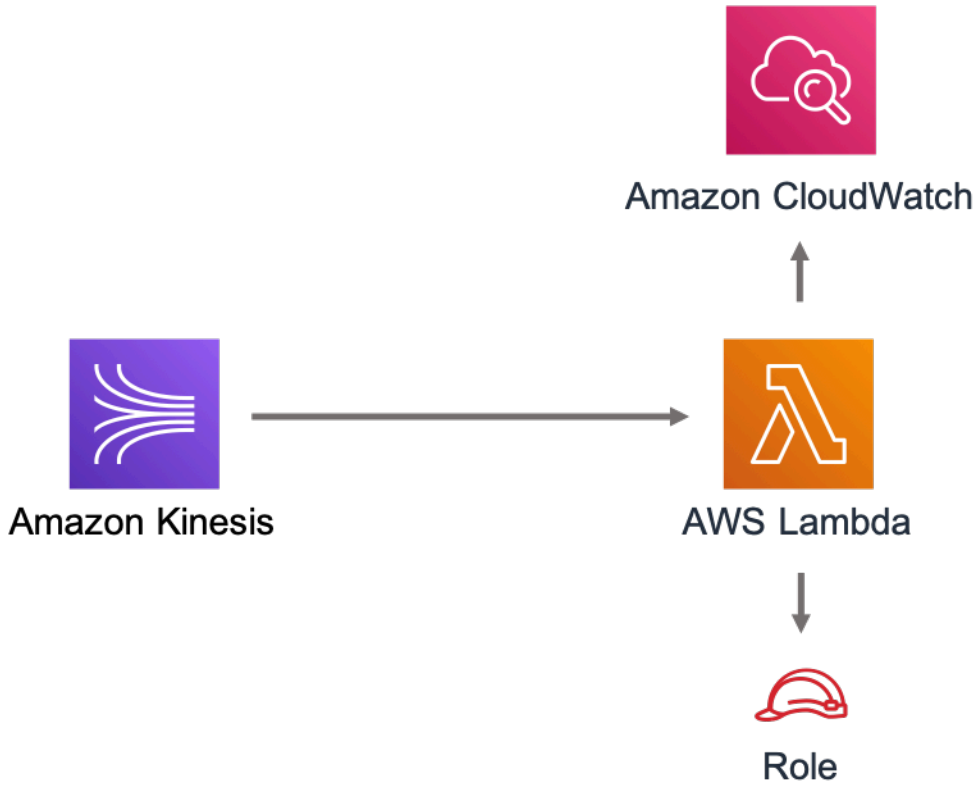
Amazon Kinesis Stream

- Configurar a função do IAM de acesso de menor privilégio para Kinesis Stream.
- Ative a criptografia do lado do servidor para o Kinesis Stream usando a chave KMS gerenciada pela AWS.
- Implante os Alarmes do CloudWatch de práticas recomendadas para o Kinesis Stream.

Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Ativar rastreamento do X-Ray.
- Ativar recursos de manipulação de falhas: habilitar bisect na função Erro; definir a Idade Máxima de Registro padrão (24 horas); definir Máximo de Tentativas de Repetição (500) padrão; e implantar a fila de letras mortas SQS como destino em caso de falha.
- SET DEFAULT
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-kinesisstreams-lambda

aws-lambda-dynamodb

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_lambda_dynamodb`
TypeScript	`@aws-solutions-constructs/aws-lambda-dynamodb`
Java	`software.amazon.awsconstructs.services.lambdadynamodb`

Overview

Este AWS Solutions Construct implementa a função do AWS Lambda e a tabela do Amazon DynamoDB com permissões de menor privilégio.

Aqui está uma definição de padrão implantável mínima no TypeScript:

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

const props: LambdaToDynamoDBProps = {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    }
};

new LambdaToDynamoDB(this, 'test-lambda-dynamodb-stack', props);
```

Initializer

```
new LambdaToDynamoDB(scope: Construct, id: string, props: LambdaToDynamoDBProps);
```

Parâmetros

- escopoConstruct
- idstring
- propsLambdaToDynamoDBProps

Aderetos de criação de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	lambda.Function	Instância existente do objeto Lambda Function, fornecendo tanto isso elambdaFunctionProps causará um erro.
LambdaFunctionProps?	lambda.FunctionProps	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se umexistingLambdaObj é fornecido.
DynamotableProps?	dynamodb.TableProps	Os adereços fornecidos pelo usuário opcionais para substituir os adereços padrão da Tabela do DynamoDB
ExistingTableobj?	dynamodb.Table	Instância existente do objeto de tabela do DynamoDB, fornecendo

Nome	Tipo	Descrição
		tanto isso quanto`dynamoTableProps` causará um erro.
TablePermissions?	`string`	Permissões de tabela opcionais a serem concedidas à função do Lambda. Uma das seguintes opções podem ser especificadas:`All`,`Read`,`ReadWrite`, ou`Write`.
TableEnvironmentVariableName?	`string`	Nome opcional para o conjunto de variáveis de ambiente de tabela do DynamoDB para a função Lambda.
ExistingVPC?	`ec2.IVpc`	Uma VPC opcional existente na qual esse padrão deve ser implantado. Quando implantada em uma VPC, a função do Lambda usará ENIs na VPC para acessar recursos de rede e um Gateway Endpoint será criado na VPC para Amazon DynamoDB. Se uma VPC existente for fornecida, o`deployVpc` A propriedade não pode ser`true`. Isso usa`ec2.IVpc`para permitir que os clientes forneçam VPCs que existem fora da pilha usando o`ec2.Vpc.fromLookup()` Método do.

Nome	Tipo	Descrição
VPCProps?	`ec2.VpcProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da nova VPC.`enableDnsHostnames`,`enableDnsSupport`,`natGateways`, `esubnetConfiguration` são definidos pelo padrão, portanto, quaisquer valores para essas propriedades fornecidas aqui serão substituídos. Se`deployVpc` não é`true`então essa propriedade será ignorada.

Nome	Tipo	Descrição
Implementar VPC?	`boolean`	Como criar uma nova VPC com base em`vpcProps`no qual implantar esse padrão. Definir isso como true implantará a VPC mínima e mais privada para executar o padrão: • Uma sub-rede isolada em cada zona de disponibi lidade usada pelo programa CDK • `enableDnsHostnames` `eenableDns` `Support` serão ambos definidos como`true` Se esta propriedade for`true`, então`existingVpc` não pode ser especificado. Padronizado como `false`.

Propriedades de padrão

Nome	Tipo	Descrição
DynamoTable	`dynamodb.Table`	Retorna uma instância da tabela do DynamoDB criada pelo padrão.
lambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.

Nome	Tipo	Descrição
VPC?	`ec2.IVpc`	Retorna uma interface na VPC usada pelo padrão (se houver). Esta pode ser uma VPC criada pelo padrão ou pela VPC fornecida ao construtor de padrões.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

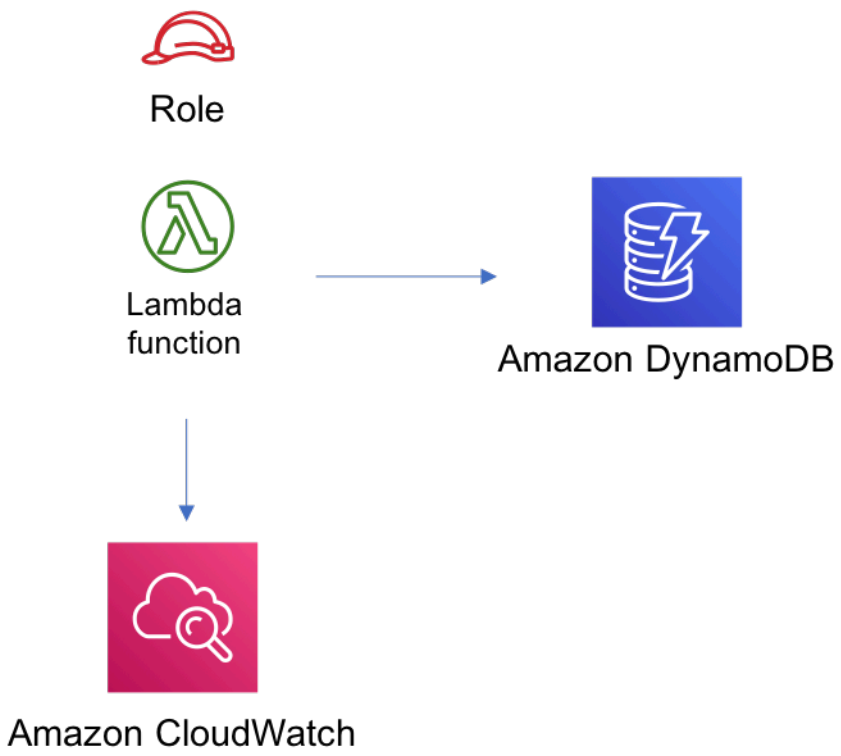
Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Ativar rastreamento do X-Ray.
- Definição de variáveis de ambiente:
  - `DDB_TABLE_NAME` (padrão)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Amazon DynamoDB

- Defina o modo de faturamento da Tabela do DynamoDB como On-Demand (Pagamento por solicitação).
- Habilite a criptografia do lado do servidor para a tabela do DynamoDB usando a chave KMS gerenciada pela AWS.
- Cria uma chave de partição chamada 'id' para a tabela do DynamoDB.
- Manter a tabela ao excluir a pilha do CloudFormation.
- Permita backups contínuos e recuperação point-in-time.

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

	@aws -solutions-constructs/aws-lambda-dynamodb

aws-lambda-elasticsearch-kibana

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_lambda_elasticsearch_kibana`
TypeScript	`@aws-solutions-constructs/aws-lambda-elasticsearch-kibana`
Java	`software.amazon.awsconstructs.services.lambdaelasticsearchkibana`

Overview

Este AWS Solutions Construct implementa uma função do AWS Lambda e um domínio do Amazon Elasticsearch Service com permissões menos privilegiadas.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { LambdaToElasticSearchAndKibana } from '@aws-solutions-constructs/aws-lambda-elasticsearch-kibana';
import { Aws } from "@aws-cdk/core";

const lambdaProps: lambda.FunctionProps = {
    runtime: lambda.Runtime.NODEJS_14_X,
    // This assumes a handler function in lib/lambda/index.js
    code: lambda.Code.fromAsset(`${__dirname}/lambda`),
    handler: 'index.handler'
};

new LambdaToElasticSearchAndKibana(this, 'test-lambda-elasticsearch-kibana', {
    lambdaFunctionProps: lambdaProps,
    domainName: 'test-domain',
    // TODO: Ensure the Cognito domain name is globally unique
    cognitoDomainName: 'globallyuniquedomain' + Aws.ACCOUNT_ID;
});
```

Initializer

```
new LambdaToElasticSearchAndKibana(scope: Construct, id: string, props:
 LambdaToElasticSearchAndKibanaProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`LambdaToElasticSearchAndKibanaProps`

Estrutura de construção de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo tanto isso `elambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.
EsdomainProps?	`elasticsearch.CfnDomainProps`	O usuário opcional forneceu adereços para substituir os adereços padrão do Amazon Elasticsearch Service

Nome	Tipo	Descrição
domainName	`string`	Nome de domínio para o Cognito e o Amazon Elasticsearch Service
CognitoDomainName?	`string`	Nome de domínio do Cognito opcional. Se fornecido, ele será usado para o domínio do Cognito, e`domainName` será usado para o domínio Elasticsearch.
CreateCloudWatchAlms	`boolean`	Criar alarmes recomendados do CloudWatch.
DomainEndPointEnvironmentVariableName?	`string`	Nome opcional para o conjunto de variáveis de ambiente de endpoint de domínio ElasticSearch para a função Lambda.

Propriedades do padrão

Nome	Tipo	Descrição
CloudwatchAlarm?	`cloudwatch.Alarm[]`	Retorna uma lista de um ou mais alarmes do CloudWatch criados pelo padrão.
ElasticSearchDomain	`elasticsearch.CfnDomain`	Retorna uma instância do domínio Elasticsearch criado pelo padrão.
ElasticSearchDomainRole	`iam.Role`	Retorna uma instância da função do IAM criada pelo

Nome	Tipo	Descrição
		padrão para o domínio do Elasticsearch.
IdentityPool	`cognito.CfnIdentityPool`	Retorna uma instância do pool de identidades do Cognito criado pelo padrão.
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.
userPool	`cognito.UserPool`	Retorna uma instância do grupo de usuários do Cognito criado pelo padrão.
UserPoolCli	`cognito.UserPoolClient`	Retorna uma instância do cliente do grupo de usuários do Cognito criado pelo padrão.

Função Lambda

Esse padrão requer uma função do Lambda que possa postar dados no serviço Elasticsearch a partir do stream do DynamoDB. Uma função de exemplo é fornecidaAqui.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Habilitar rastreamento do X-Ray.
- Definir variáveis de ambiente:
  - `DOMAIN_ENDPOINT` (padrão)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

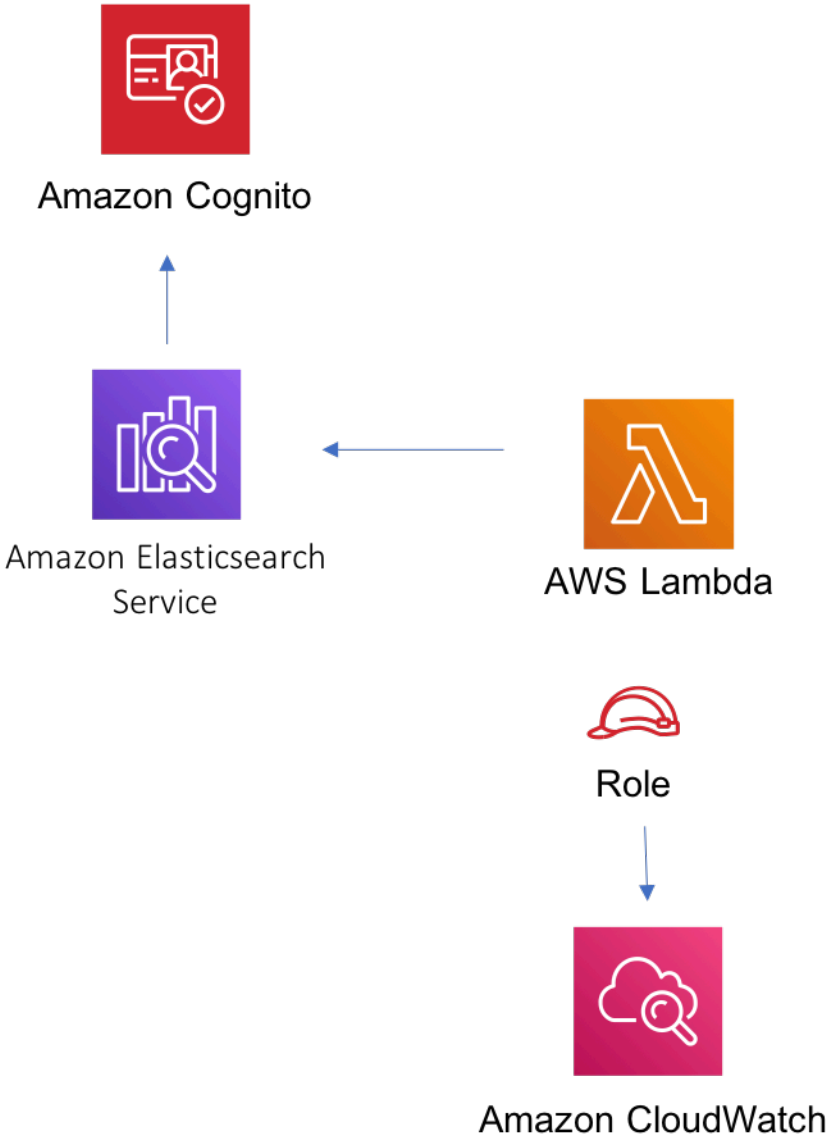
Amazon Cognito

- Defina a política de senha para grupos de usuários.
- Imponha o modo de segurança avançado para grupos de usuários.

Amazon Elasticsearch Service

- Implante as melhores práticas Alarmes do CloudWatch para o domínio do Elasticsearch.
- Proteja o acesso ao painel Kibana com grupos de usuários do Cognito.
- Ative a criptografia do lado do servidor para o domínio do Elasticsearch usando a chave KMS gerenciada pela AWS.
- Habilite a criptografia de nó a nó para o domínio do Elasticsearch.
- Configurar o cluster para o domínio do Amazon ES.

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-lambda-elasticsearch-kibana

aws-lambda-s3

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_lambda_s3`
TypeScript	`@aws-solutions-constructs/aws-lambda-s3`
Java	`software.amazon.awsconstructs.services.lambdas3`

Overview

Este AWS Solutions Construct implementa uma função do AWS Lambda conectada a um bucket do Amazon S3.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { LambdaToS3 } from '@aws-solutions-constructs/aws-lambda-s3';

new LambdaToS3(this, 'LambdaToS3Pattern', {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
```

```
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    }
});
```

Initializer

```
new LambdaToS3(scope: Construct, id: string, props: LambdaToS3Props);
```

Parâmetros

- escopoConstruct
- idstring
- propsLambdaToS3Props

Adereços de construção de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	lambda.Function	Instância existente do objeto Lambda Function, fornecendo tanto isso elambdaFunctionProps causará um erro.
LambdaFunctionProps?	lambda.FunctionProps	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se umexistingLambdaObj é fornecido.
ExistingBucketoBJ?	s3.IBucket	Instância existente do objeto S3 Bucket. Se isso for fornecido, então também

Nome	Tipo	Descrição
		fornecendo`bucketProps` é um erro.
Baldes?	`s3.BucketProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão do bucket. Ignorado se um`existingBucketObj` é fornecido.
Permissões de Bucket?	`string[]`	Permissões de bucket opcionais para conceder à função do Lambda. Um ou mais itens a seguir podem ser especific ados:`Delete`,`Put`,`Read`,`ReadWrite`,`Write`.
ExistingVPC?	`ec2.IVpc`	Uma VPC opcional existente na qual esse padrão deve ser implantado. Quando implantada em uma VPC, a função Lambda usará ENIs na VPC para acessar recursos de rede e um ponto final de interface será criado na VPC para Amazon SQS. Se uma VPC existente for fornecida, o`deployVpc` A proprieda de não pode ser`true`. Isso usa`ec2.IVpc`para permitir que os clientes forneçam VPCs que existem fora da pilha usando o`ec2.Vpc.fromLookup()` Método do.

Nome	Tipo	Descrição
Implementar VPC?	`boolean`	Como criar uma nova VPC com base em`vpcProps`no qual implantar esse padrão. Configuração como`true`implantará a VPC mínima e mais privada para executar o padrão: • Uma sub-rede isolada em cada zona de disponibi lidade usada pelo programa de CDK. • `enableDnsHostnames` e`enableDns Support` serão ambos definidos como`true`. Se esta propriedade for`true`, então`existingVpc` não pode ser especificado. Padronizado como `false`.

Nome	Tipo	Descrição
VPCProps?	`ec2.VpcProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da nova VPC.`enableDns Hostnames` ,`enableDns Support` ,`natGatewa ys` `esubnetCon figuration` são definidos pelo padrão, portanto, quaisquer valores para essas propriedades fornecida s aqui serão substituí dos. Se`deployVpc` não é`true`então essa propriedade será ignorada.
BucketEnvironmentVariableNa me?	`string`	Nome opcional para o conjunto de variáveis de ambiente do bucket S3 para a função Lambda.

Propriedades de padrão

Nome	Tipo	Descrição
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.
S3Bucket?	`s3.Bucket`	Retorna uma instância do bucket S3 criado pelo padrão.

Nome	Tipo	Descrição
S3loggingBucket?	`s3.Bucket`	Retorna uma instância do bucket de log criado pelo padrão para o bucket S3.
VPC?	`ec2.IVpc`	Retorna uma instância da VPC usada pelo padrão (se houver). Esta pode ser uma VPC criada pelo padrão ou pela VPC fornecida ao construtor de padrões.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Ativar rastreamento do X-Ray
- Definição de variáveis de ambiente:
  - `S3_BUCKET_NAME` (padrão)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Amazon S3 Bucket

- Configurar o log de acesso para o S3 Bucket.
- Ative a criptografia do lado do servidor para o S3 Bucket usando a chave KMS gerenciada pela AWS.
- Ative o controle de versão para o bucket do S3.
- Não permitir acesso público para o S3 Bucket.
- Mantenha o bucket do S3 ao excluir a pilha do CloudFormation.
- Aplique a criptografia de dados em trânsito

- Aplica a regra de ciclo de vida para mover versões de objetos não atuais para o armazenamento do Glacier após 90 dias.

Architecture

GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-lambda-s3

aws-lambda-ssmstringparameter

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_lambda_ssm_string_parameter`
TypeScript	`@aws-solutions-constructs/aws-lambda-ssmstringparameter`
Java	`software.amazon.awsconstructs.services.lambdassmstringparameter`

Overview

Este AWS Solutions Construct implementa a função AWS Lambda e o parâmetro String do AWS Systems Manager Parameter Store com as permissões menos privilegiadas.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
const { LambdaToSsmstringparameterProps,  LambdaToSsmstringparameter } from '@aws-solutions-constructs/aws-lambda-ssmstringparameter';

const props: LambdaToSsmstringparameterProps = {
    lambdaFunctionProps: {
      runtime: lambda.Runtime.NODEJS_14_X,
      // This assumes a handler function in lib/lambda/index.js
      code: lambda.Code.fromAsset(`${__dirname}/lambda`),
      handler: 'index.handler'
    },
    stringParameterProps: { stringValue: "test-string-value" }
};

new LambdaToSsmstringparameter(this, 'test-lambda-ssmstringparameter-stack', props);
```

Initializer

```
new LambdaToSsmstringparameter(scope: Construct, id: string, props:
 LambdaToSsmstringparameterProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`LambdaToSsmstringparameterProps`

Adereços de criação de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo tanto isso `elambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.
ExistingStringParameterobj?	`ssm.StringParameter`	Instância existente do objeto de parâmetro String SSM, fornecendo tanto isso quanto`stringParameterProps` causará um erro.

Nome	Tipo	Descrição
StringParameter Props?	`ssm.StringParameterProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para o parâmetro String SSM. Se `existingStringParameterObj` não estiver definido, `stringParameterProps` é necessário. O único suportado `ssm.StringParameterProps.type` é `STRING` se um valor diferente for fornecido, ele será substituído.
StringParameterEnvironmentVariableName?	`string`	Nome opcional para o conjunto de variáveis de ambiente do parâmetro String SSM para a função Lambda.

Nome	Tipo	Descrição
ExistingVPC?	`ec2.IVpc`	Uma VPC opcional existente na qual esse padrão deve ser implantado. Quando implantada em uma VPC, a função Lambda usará ENIs na VPC para acessar recursos de rede e um ponto final de interface será criado na VPC para AWS Systems Manager Parâmetro. Se uma VPC existente for fornecida , o`deployVpc` A proprieda de não pode ser`true`. Isso usa o`ec2.IVpc`para permitir que os clientes forneçam VPCs que existem fora da pilha usando o`ec2.Vpc.fromLookup()` Método do.

Nome	Tipo	Descrição
VPCProps?	`ec2.VpcProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da nova VPC.`enableDns Hostnames` ,`enableDns Support` ,`natGatewa ys` `esubnetCon figuration` são definidos pelo padrão, portanto, quaisquer valores para essas propriedades fornecida s aqui serão substituí dos. Se`deployVpc` não é`true`Então essa propriedade será ignorada.

Nome	Tipo	Descrição
Implementar VPC?	`boolean`	Criar uma nova VPC com base em`vpcProps`no qual implantar esse padrão. Configuração deste como`true`implantará a VPC mínima e mais privada para executar o padrão: • Uma sub-rede isolada em cada zona de disponibi lidade usada pelo programa CDK. • `enableDnsHostnames` e`enableDns Support` serão ambos definidos como`true`. Se essa propriedade é definida como`true`, então`existingVpc` não pode ser especificado. Padronizado como `false`.
Permissões de StringPar ameter?	`string`	Permissões opcionais do parâmetro String SSM para conceder à função Lambda. Uma das opções a seguir pode ser especific ada:`Read`,`ReadWrite` .

Propriedades de padrão

Nome	Tipo	Descrição
LambdaFunction	`lambda.Function`	Retorna uma instância de`lambda.Function` criado pela construção.
StringParameter	`ssm.StringParameter`	Retorna uma instância de`ssm.StringParameter` criado pela construção.
VPC?	`ec2.IVpc`	Retorna uma interface na VPC usada pelo padrão (se houver). Esta pode ser uma VPC criada pelo padrão ou pela VPC fornecida ao construtor de padrões.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Função do AWS Lambda

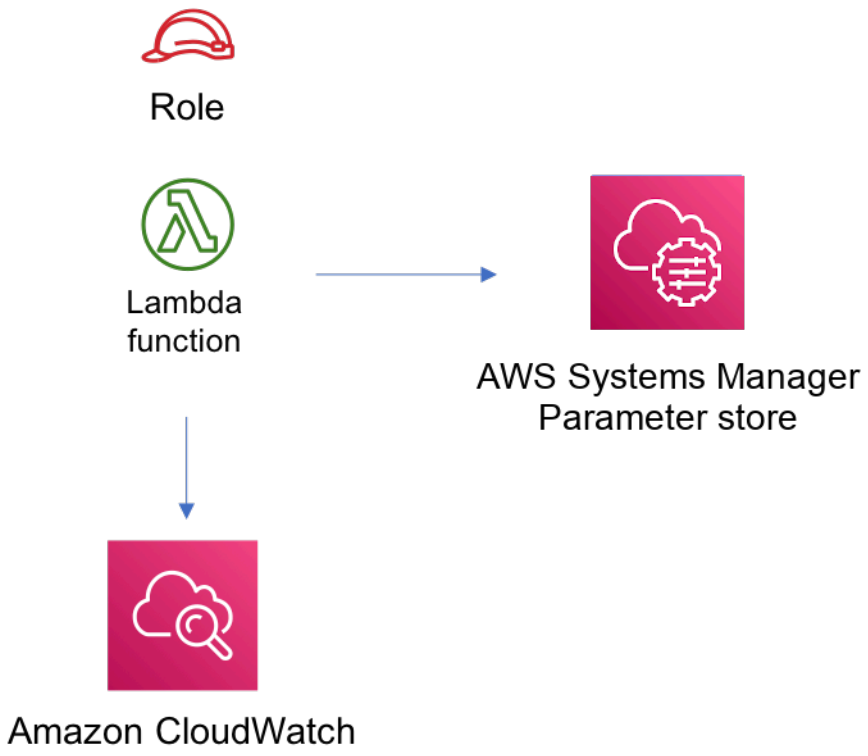
- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Ativar rastreamento do X-Ray.
- Definir variáveis de ambiente:
  - `SSM_STRING_PARAMETER_NAME` (padrão)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

String de repositório de parâmetros do Amazon AWS Systems Manager

- Habilite o acesso somente leitura para a função do AWS Lambda associada.
- Cria um novo parâmetro String SSM com os valores fornecidos.

- Mantenha o parâmetro String SSM ao excluir a pilha do CloudFormation.

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-lambda-ssm stringparameter

aws-lambda-sagemakerendpoint

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_lambda_sagemakerendpoint`
TypeScript	`@aws-solutions-constructs/aws-lambda-sagemakerendpoint`
Java	`software.amazon.awsconstructs.services.lambdasagemakerendpoint`

Overview

Este AWS Solutions Construct implementa uma função do AWS Lambda conectada a um endpoint do Amazon Sagemaker.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { Duration } from '@aws-cdk/core';
import * as lambda from '@aws-cdk/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,
    // This assumes a handler function in lib/lambda/index.py
    code: lambda.Code.fromAsset(`${__dirname}/lambda`),
    handler: 'index.handler',
    timeout: Duration.minutes(5),
    memorySize: 128,
  },
};

new LambdaToSagemakerEndpoint(this, 'LambdaToSagemakerEndpointPattern',
 constructProps);
```

Initializer

```
new LambdaToSagemakerEndpoint(scope: Construct, id: string, props:
 LambdaToSagemakerEndpointProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`LambdaToSagemakerEndpointProps`

Adereços de criação de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo tanto isso e `lambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para

Nome	Tipo	Descrição
		substituir as propriedades padrão da função Lambda.
ExistingSagemakerEndpointoBJ?	`sagemaker.CfnEndpoint`	Um Sagemaker Enpoint opcional existente para ser usado. Fornecendo tanto isso quanto`endpointProps` causará um erro.
ModelProps?	`sagemaker.CfnModelProps` | `any`	Propriedades fornecidas pelo usuário para substituir as propriedades padrão do Modelo Sagemaker. No mínimo`modelProps.primaryContainer` deve ser fornecido para criar um modelo. Por padrão, o padrão criará uma função com as permissões mínimas necessárias, mas o cliente pode fornecer uma função personalizada com recursos adicionais usando`modelProps.executionRoleArn`.
EndpointConfigProps?	`sagemaker.CfnEndpointConfigProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da configuração do Sagemaker Endpoint.
EndpointProps?	`sagemaker.CfnEndpointProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão do Sagemaker Endpoint.

Nome	Tipo	Descrição
ExistingVPC?	`ec2.IVpc`	Uma VPC existente opcional na qual essa construçã o deve ser implantada. Quando implantados em uma VPC, a função Lambda e o Sagemaker Endpoint usarão ENIs na VPC para acessar recursos de rede. Um ponto final de interface será criado na VPC para Amazon Sagemaker Runtime e Amazon S3 VPC Endpoint. Se uma VPC existente for fornecida, o`deployVpc` A propriedade não pode ser`true`.
VPCProps?	`ec2.VpcProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da nova VPC.`enableDns Hostnames` ,`enableDns Support` ,`natGatewa ys` `esubnetCon figuration` são definidos pela construção, portanto, quaisquer valores para essas propriedades fornecidas aqui serão substituídos. Se`deployVpc` não é`true`, então essa propriedade será ignorada.

Nome	Tipo	Descrição
Implementar VPC?	`boolean`	Criar uma nova VPC com base em`vpcProps`no qual implantar esse padrão. Configuração como`true`implantará a VPC mínima e mais privada para executar o padrão: • Uma sub-rede isolada em cada zona de disponibi lidade usada pelo programa CDK. • `enableDnsHostnames` e`enableDns Support` serão ambos definidos como`true`. Se essa propriedade for definida como`true`, então`existingVpc` não pode ser especificado. Padronizado como `false`.
SageMakerEnvironme ntVariableName?	`string`	Nome opcional para o conjunto de variáveis de ambiente de endpoint SageMaker para a função Lambda.

Propriedades de padrão

Nome	Tipo	Descrição
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.
SageMakerEndpoint	`sagemaker.CfnEndpoint`	Retorna uma instância do Sagemaker Endpoint criado pelo padrão.
SageMakerEndPointConfig?	`sagemaker.CfnEndpointConfig`	Retorna uma instância do SageMaker EndpointConfig criado pelo padrão, se `existingSagemakerEndpointObj` não foi fornecido.
Modelo Sagemaker?	`sagemaker.CfnModel`	Retorna uma instância do Modelo Sagemaker criado pelo padrão, se `existingSagemakerEndpointObj` não foi fornecido.
VPC?	`ec2.IVpc`	Retorna uma instância da VPC criada pelo padrão, se `deployVpc` é `true`, ou se `existingVpc` é fornecido.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Função do AWS Lambda

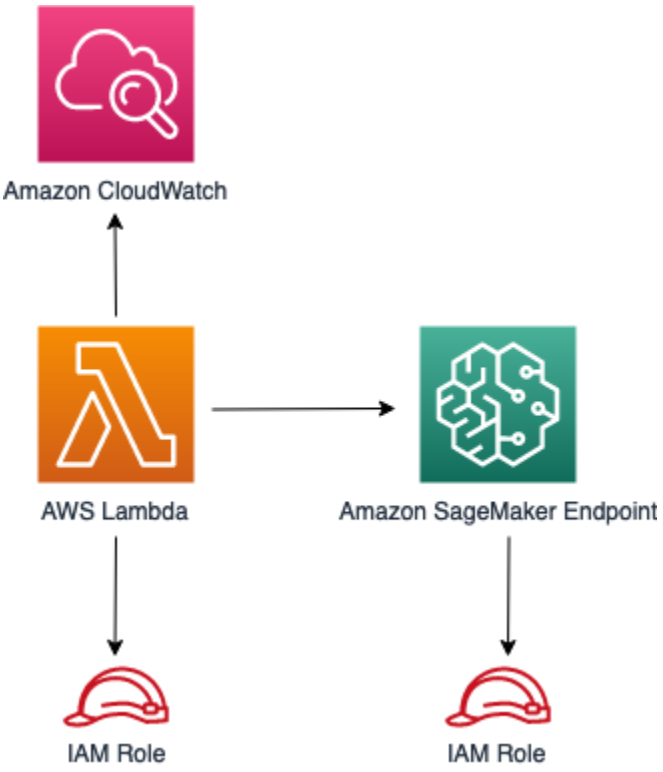
- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.

- Permita que a função chame o ponto final do Sagemaker para Inferências.
- Configure a função para acessar recursos na VPC, onde o endpoint do Sagemaker é implantado.
- Ativar Rastreamento do X-Ray
- Definição de variáveis de ambiente:
  - `SAGEMAKER_ENDPOINT_NAME` (padrão)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Endpoint do Amazon Sage

- Configure privilégios limitados para criar recursos do Sagemaker.
- Implante o modelo Sagemaker, o EndPointConfig e o endpoint.
- Configure o endpoint do Sagemaker a ser implantado em uma VPC.
- Implante a interface VPC Endpoint e Sagemaker Runtime VPC.

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-lambda-sagemakerendpoint

aws-lambda-secretsmanager

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àSemantic versionModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_lambda_secretsmanager`
TypeScript	`@aws-solutions-constructs/aws-lambda-secretsmanager`
Java	`software.amazon.awsconstructs.services.lambdasecretsmanager`

Overview

Este AWS Solutions Construct implementa a função do AWS Lambda e o segredo do AWS Secrets Manager com as permissões menos privilegiadas.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
const { LambdaToSecretsmanagerProps,  LambdaToSecretsmanager } from '@aws-solutions-
constructs/aws-lambda-secretsmanager';

const props: LambdaToSecretsmanagerProps = {
    lambdaFunctionProps: {
      runtime: lambda.Runtime.NODEJS_14_X,
      // This assumes a handler function in lib/lambda/index.js
      code: lambda.Code.fromAsset(`${__dirname}/lambda`),
      handler: 'index.handler'
    },
};

new LambdaToSecretsmanager(this, 'test-lambda-secretsmanager-stack', props);
```

Initializer

```
new LambdaToSecretsmanager(scope: Construct, id: string, props:
 LambdaToSecretsmanagerProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`LambdaToSecretsmanagerProps`

Adereços de criação de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecend o tanto isso `elambdaFun`

Nome	Tipo	Descrição
		`ctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	O usuário forneceu adereços para substituir os adereços padrão para a função Lambda.
SecretProps?	`secretsmanager.SecretProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para o Secrets Manager.
ExistingSecrettoBJ?	`secretsmanager.Secret`	Instância existente do objeto secreto Secrets Manager dos, Se isso estiver definido, a propriedade`secretProps` é ignorado.
GrantWriteAccess?	`boolean`	Acesso de gravação opcional ao segredo para a função Lambda (somente leitura por padrão).
Nome do secretárioVironmentVariableName?	`string`	Nome opcional para o conjunto de variáveis de ambiente secreto do Secrets Manager para a função Lambda.

Nome	Tipo	Descrição
ExistingVPC?	`ec2.IVpc`	Uma VPC opcional existente na qual esse padrão deve ser implantado. Quando implantada em uma VPC, a função Lambda usará ENIs na VPC para acessar recursos de rede e um ponto final de interface será criado na VPC para AWS Secrets Manager. Se uma VPC existente for fornecida, o `deployVpc` A propriedade não pode ser `true`. Isso usa `ec2.IVpc` para permitir que os clientes forneçam VPCs que existem fora da pilha usando o `ec2.Vpc.fromLookup()` Método do.
VPCProps?	`ec2.VpcProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da nova VPC. `enableDnsHostnames`, `enableDnsSupport`, `natGateways`, e `subnetConfiguration` são definidos pelo padrão, portanto, quaisquer valores para essas propriedades fornecidas aqui serão substituídos. Se `deployVpc` não é `true` então essa propriedade será ignorada.

Nome	Tipo	Descrição
Implementar VPC?	`boolean`	Criar uma nova VPC com base no`vpcProps`no qual implantar esse padrão. Configurando isso como`true`implantará a VPC mínima e mais privada para executar o padrão: • Uma sub-rede isolada em cada zona de disponibi lidade usada pelo programa CDK • `enableDnsHostnames` e`enableDns Support` serão ambos definidos como`true` Se esta propriedade for`true`, então`existingVpc` não pode ser especificado. Padronizado como `false`.

Propriedades de padrão

Nome	Tipo	Descrição
LambdaFunction	`lambda.Function`	Retorna uma instância de`lambda.Function` criado pela construção.
segredo	`secretsmanager.Secret`	Retorna uma instância de`secretsmanager.Secret` criado pela construção.

Nome	Tipo	Descrição
VPC?	`ec2.IVpc`	Retorna uma interface na VPC usada pelo padrão (se houver). Esta pode ser uma VPC criada pelo padrão ou pela VPC fornecida ao construtor de padrões.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

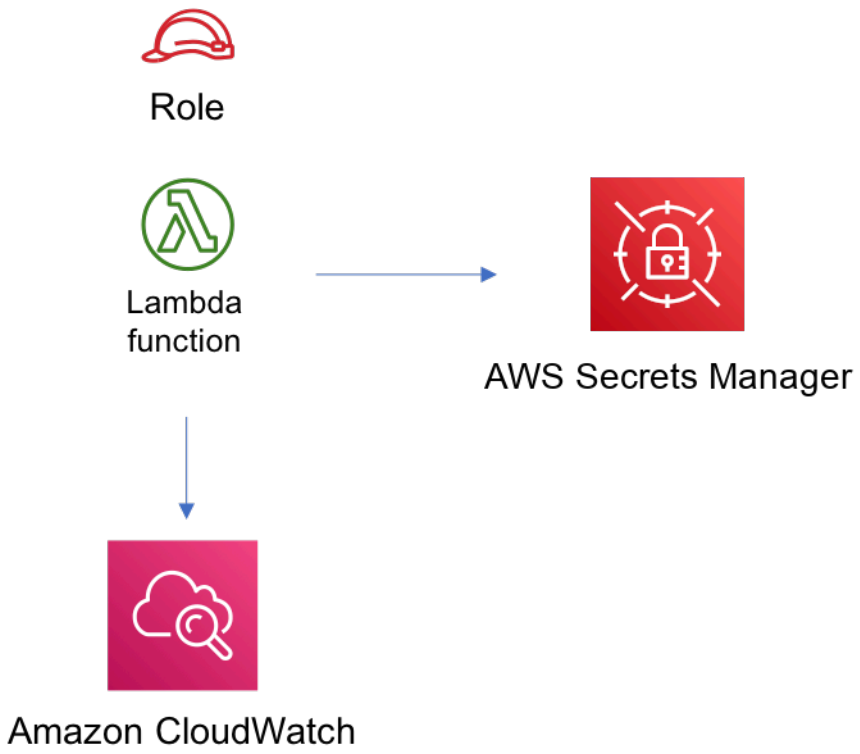
Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Ativar rastreamento do X-Ray.
- Definir variáveis de ambiente:
  - (padrão) SECRET_ARN contendo o ARN do segredo como retorno pelo CDKSecretArnA propriedade
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Segredo do Amazon Secrets Manager

- Habilitar acesso somente leitura para a função do AWS Lambda associada
- Habilitar a criptografia do lado do servidor usando uma chave KMS padrão para a conta e a região
- Cria um novo segredo:
  - (padrão) nome aleatório
  - valor aleatório (padrão)
- Manter o segredo ao excluir a pilha do CloudFormation

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-lambda-secretsmanager

aws-lambda-sns

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_lambda_sns`
TypeScript	`@aws-solutions-constructs/aws-lambda-sns`
Java	`software.amazon.awsconstructs.services.lambdasns`

Overview

Este AWS Solutions Construct implementa uma função do AWS Lambda conectada a um tópico do Amazon SNS.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { LambdaToSns, LambdaToSnsProps } from "@aws-solutions-constructs/aws-lambda-
sns";

new LambdaToSns(this, 'test-lambda-sns', {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    }
});
```

Initializer

```
new LambdaToSns(scope: Construct, id: string, props: LambdaToSnsProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`LambdaToSnsProps`

Adereços de criação de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo tanto isso e`lambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.
ExistingTopicobj?	`sns.Topic`	Instância existente do objeto Tópico SNS, fornecendo tanto isso quanto`topicProps` causará um erro.
TopicProps?	`sns.TopicProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão do tópico SNS.
ExistingVPC?	`ec2.IVpc`	Uma VPC opcional existente na qual esse padrão deve ser implantado. Quando

Nome	Tipo	Descrição
		implantada em uma VPC, a função Lambda usará ENIs na VPC para acessar recursos de rede e um ponto final de interface será criado na VPC para Amazon SQS. Se uma VPC existente for fornecida, a proprieda de`deployVpc` A proprieda de não pode ser`true`. Isso usa`ec2.IVpc`para permitir que os clientes forneçam VPCs que existem fora da pilha usando o`ec2.Vpc.fromLookup()` Método do.

Nome	Tipo	Descrição
Implementar VPC?	`boolean`	Como criar uma nova VPC com base em`vpcProps`no qual implantar esse padrão. Configuração como`true`implantará a VPC mínima e mais privada para executar o padrão: • Uma sub-rede isolada em cada zona de disponibi lidade usada pelo programa de CDK. • `enableDnsHostnames` e`enableDns Support` serão ambos definidos como`true`. Se esta propriedade for`true`, então`existingVpc` não pode ser especificado. Padronizado como `false`.

Nome	Tipo	Descrição
VPCProps?	`ec2.VpcProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da nova VPC.`enableDns Hostnames` ,`enableDns Support` ,`natGatewa ys` `esubnetCon figuration` são definidos pelo padrão, portanto, quaisquer valores para essas propriedades fornecida s aqui serão substituí dos. Se`deployVpc` não é`true`então essa propriedade será ignorada.
TopicarnEnvironmentVariable Name?	`string`	Nome opcional para o conjunto de variáveis de ambiente ARN do tópico do SNS para a função do Lambda.
TopicNameEnvironme ntVariableName?	`string`	Nome opcional para o conjunto de variáveis de ambiente de nome de tópico SNS para a função Lambda.

Propriedades de padrão

Nome	Tipo	Descrição
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.
snsTopic	`sns.Topic`	Retorna uma instância do tópico SNS criado pelo padrão.
VPC?	`ec2.IVpc`	Retorna uma instância da VPC usada pelo padrão (se houver). Esta pode ser uma VPC criada pelo padrão ou pela VPC fornecida ao construtor de padrões.

Configurações padrão

A implementação imediata da Construct sem qualquer substituição definirá os seguintes padrões:

Função do AWS Lambda

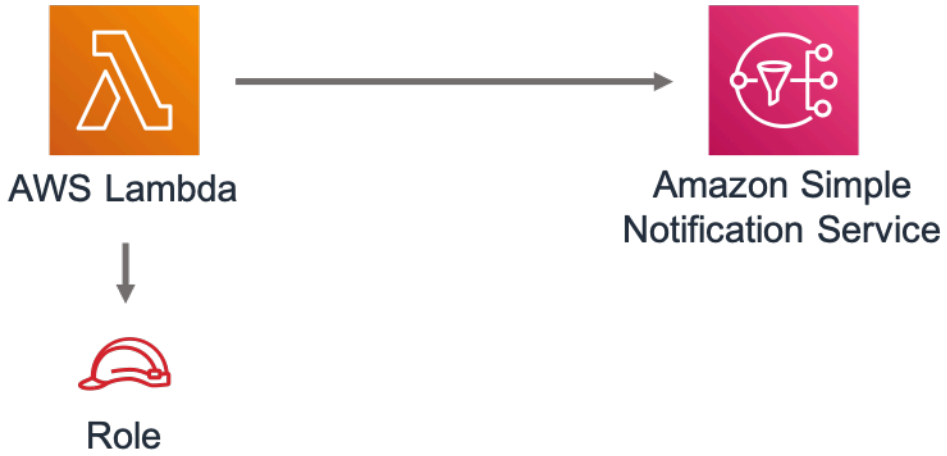
- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Ativar rastreamento do X-Ray.
- Definição de variáveis de ambiente:
  - `SNS_TOPIC_NAME` (padrão)
  - `SNS_TOPIC_ARN` (padrão)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Tópico do Amazon SNS

- Configurar permissões de acesso de menor privilégio para o tópico SNS.

- Ative a criptografia do lado do servidor usando a chave KMS gerenciada pela AWS.
- Aplique a criptografia de dados em trânsito

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-lambda-sns

aws-lambda-sqs

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_lambda_sqs`
TypeScript	`@aws-solutions-constructs/aws-lambda-sqs`
Java	`software.amazon.awsconstructs.services.lambdasqs`

Overview

Este AWS Solutions Construct implementa uma função do AWS Lambda conectada a uma fila do Amazon SQS.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { LambdaToSqs, LambdaToSqsProps } from "@aws-solutions-constructs/aws-lambda-sqs";

new LambdaToSqs(this, 'LambdaToSqsPattern', {
   lambdaFunctionProps: {
       runtime: lambda.Runtime.NODEJS_14_X,
       // This assumes a handler function in lib/lambda/index.js
       code: lambda.Code.fromAsset(`${__dirname}/lambda`),
       handler: 'index.handler'
   }
});
```

Initializer

```
new LambdaToSqs(scope: Construct, id: string, props: LambdaToSqsProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`LambdaToSqsProps`

Adereços de construção de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Uma função opcional existente do Lambda a ser usada em vez da função padrão. Fornecendo tanto isso quanto`lambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda.
ExistingQueueobj?	`sqs.Queue`	Uma fila SQS opcional existente a ser usada em vez da fila padrão. Fornecendo tanto isso quanto`queueProps` causará um erro.
QueueProps?	`sqs.QueueProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da fila SQS.
EnableQueuePurging?	`boolean`	Se deve conceder permissões adicionais à função do

Nome	Tipo	Descrição
		Lambda, habilitando-o a limpar a fila SQS. Padronizado como `false`.
ImplantyDeadletterQueue?	`boolean`	Criar uma fila secundária a ser usada como uma dead letter queue. Padronizado como `true`.
DeadletterQueueProps?	`sqs.QueueProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para a fila de letras inativas. Usado somente se `odeployDeadLetterQueue` é definida como true.
MaxReceiveCount?	`number`	O número de vezes que uma mensagem pode ser desenfileirada sem sucesso antes de ser movida para a fila de mensagens mortas. Padronizado como 15.

Nome	Tipo	Descrição
ExistingVPC?	`ec2.IVpc`	Uma VPC opcional existente na qual esse padrão deve ser implantado. Quando implantada em uma VPC, a função Lambda usará ENIs na VPC para acessar recursos de rede e um ponto final de interface será criado na VPC para Amazon SQS. Se uma VPC existente for fornecida, o`deployVpc` A propriedade não pode ser`true`. Uma`ec2.IVpc`é usado para permitir que os clientes forneçam VPCs que existem fora da pilha usando o`ec2.Vpc.fromLookup()` Método do.

Nome	Tipo	Descrição
Implementar VPC?	`boolean`	Criar uma nova VPC com base em`vpcProps`no qual implantar esse padrão. Configurando isso como`true`implantará a VPC mínima e mais privada para executar o padrão: • Uma sub-rede isolada em cada zona de disponibi lidade usada pelo programa CDK • `enableDnsHostnames` e`enableDns Support` serão ambos definidos como`true` Se esta propriedade for`true`, então`existingVpc` não pode ser especificado. Padronizado como `false`.

Nome	Tipo	Descrição
VPCProps?	`ec2.VpcProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da nova VPC.`enableDnsHostnames`,`enableDnsSupport`,`natGateways`, `esubnetConfiguration` são definidos pelo padrão, portanto, quaisquer valores para essas propriedades fornecidas aqui serão substituídos. Se`deployVpc` não é`true`, então essa propriedade será ignorada.
QueueEnvironmentVariableName?	`string`	Nome opcional para o conjunto de variáveis de ambiente de URL de fila SQS para a função Lambda.

Propriedades de padrão

Nome	Tipo	Descrição
DeadletterQueue?	`sqs.Queue`	Retorna uma instância da fila de letras mortas criada pelo padrão, se uma for implantada.
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.

Nome	Tipo	Descrição
SQSqueue	`sqs.Queue`	Retorna uma instância da fila SQS criada pelo padrão.
VPC?	`ec2.IVpc`	Retorna uma instância da VPC criada ou usada pelo padrão (se houver). Esta pode ser uma VPC criada pelo padrão ou uma VPC fornecida ao construtor de padrões.

Configurações padrão

A implementação imediata da Construct sem qualquer substituição definirá os seguintes padrões:

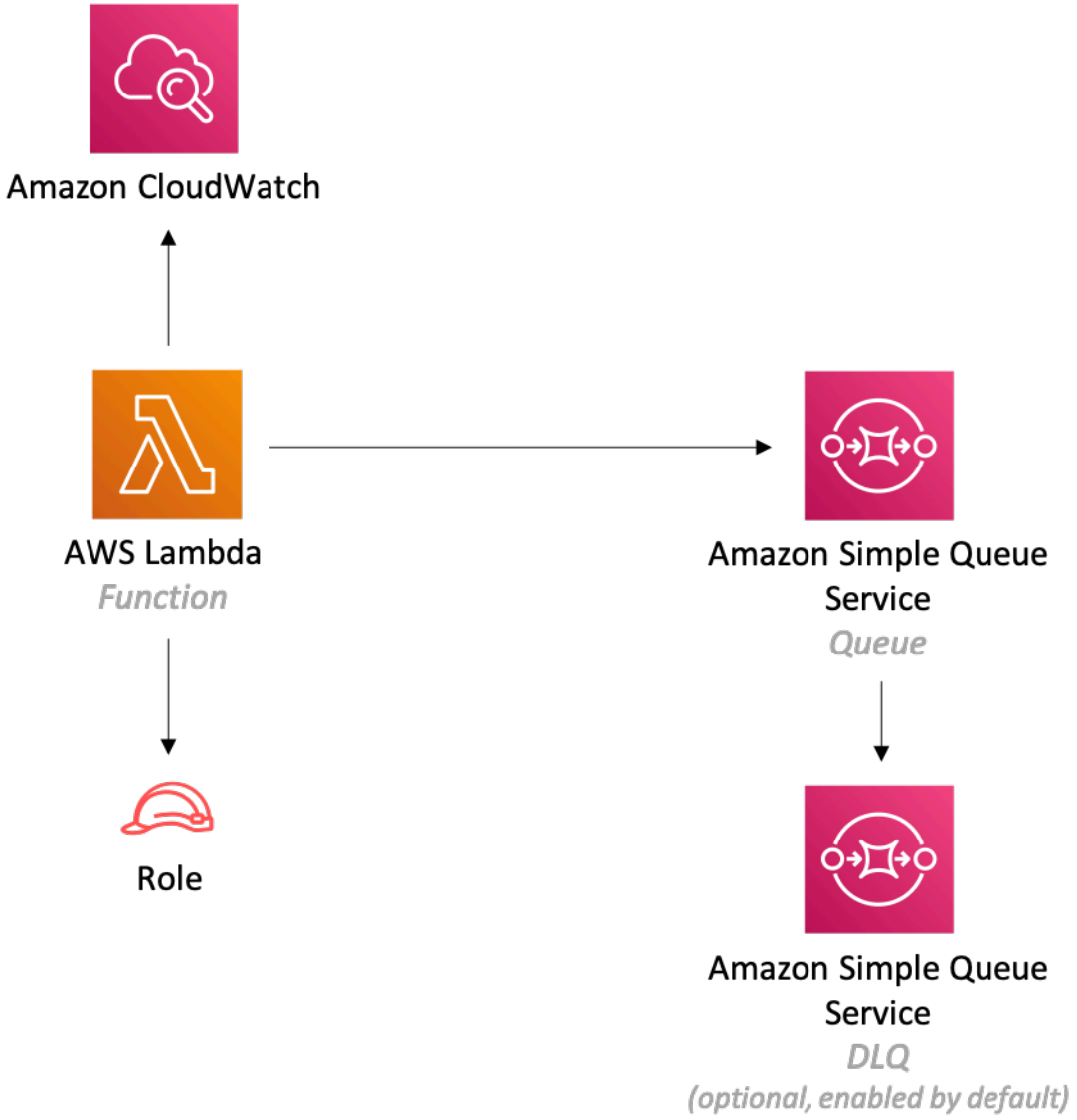
Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Permitir que a função envie apenas mensagens para a fila (a expurgação pode ser habilitada usando `oenableQueuePurge`).
- Habilitar rastreamento do X-Ray
- Definir variáveis de ambiente:
  - `SQS_QUEUE_URL`
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Fila do Amazon SQS

- Implante a dead letter queue do SQS para a fila do SQS de origem.
- Ative a criptografia no lado do servidor para a fila do SQS de origem usando a AWS Managed KMS Key.
- Aplique a criptografia de dados em trânsito

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-lambda-sqs

aws-lambda-sqs-lambda

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_lambda_sqs_lambda`
TypeScript	`@aws-solutions-constructs/aws-lambda-sqs-lambda`
Java	`software.amazon.awsconstructs.services.lambdasqslambda`

Overview

Este padrão de constrói soluções da AWS implementa (1) uma função do AWS Lambda configurada para enviar mensagens para uma fila; (2) uma fila do Amazon SQS; e (3) uma função do AWS Lambda configurada para consumir mensagens da fila.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { LambdaToSqsToLambda, LambdaToSqsToLambdaProps } from "@aws-solutions-constructs/aws-lambda-sqs-lambda";

new LambdaToSqsToLambda(this, 'LambdaToSqsToLambdaPattern', {
    producerLambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/producer-function/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda/producer-function`),
        handler: 'index.handler'
```

```
    },
    consumerLambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/consumer-function/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda/consumer-function`),
        handler: 'index.handler'
    }
});
```

Initializer

```
new LambdaToSqsToLambda(scope: Construct, id: string, props: LambdaToSqsToLambdaProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`LambdaToSqsToLambdaProps`

Adereços de construção de padrão

Nome	Tipo	Descrição
ExistingProducerLambdaobj?	`lambda.Function`	Uma função opcional existente do Lambda a ser usada em vez da função padrão para enviar mensagens para a fila. Fornecendo tanto isso quanto`producerLambdaFunctionProps` causará um erro.
ProducerLambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades

Nome	Tipo	Descrição
		padrão da função Lambda do produtor.
ExistingQueueobj?	`sqs.Queue`	Uma fila SQS opcional existente a ser usada em vez da fila padrão. Fornecendo tanto isso quanto`queueProps` causará um erro.
QueueProps?	`sqs.QueueProps`	Propriedades opcionais fornecidas pelo usuário para substituir as proprieda des padrão da fila SQS. Fornecendo tanto isso quanto`existingQueueObj` causará um erro.
Implementar DeadletterQueue?	`boolean`	Criar uma fila secundária para ser usada como uma dead letter queue. Padronizado como `true`.
DeadletterQueueProps?	`sqs.QueueProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para a fila de letras inativas. Usado somente se o`deployDeadLetterQueue` propriedade é definida como`true`.
MaxReceiveCount?	`number`	O número de vezes que uma mensagem pode ser desenfileirada sem êxito antes de ser movida para a fila de mensagens mortas. Padronizado como 15.

Nome	Tipo	Descrição
ExistingConsumerLambdaobj?	`lambda.Function`	Uma função opcional existente do Lambda a ser usada em vez da função padrão para recebição/consumo de mensagens da fila. Fornecendo tanto isso quanto`consumerLambdaFunctionProps` causará um erro.
ConsumerLambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda do consumidor.
QueueEnvironmentVariableName?	`string`	Nome opcional para o conjunto de variáveis de ambiente de URL de fila SQS para a função Lambda produtor.

Propriedades de padrão

Nome	Tipo	Descrição
ConsumerLambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda consumidor criada pelo padrão.
DeadletterQueue?	`sqs.Queue`	Retorna uma instância da fila de letras mortas criada pelo padrão, se uma for implantada.

Nome	Tipo	Descrição
ProducerLambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda produtor criada pelo padrão.
SQSqueue	`sqs.Queue`	Retorna uma instância da fila SQS criada pelo padrão.

Configurações padrão

Implementação imediata deste Construct (sem quaisquer propriedades substituídas) irá aderir aos seguintes padrões:

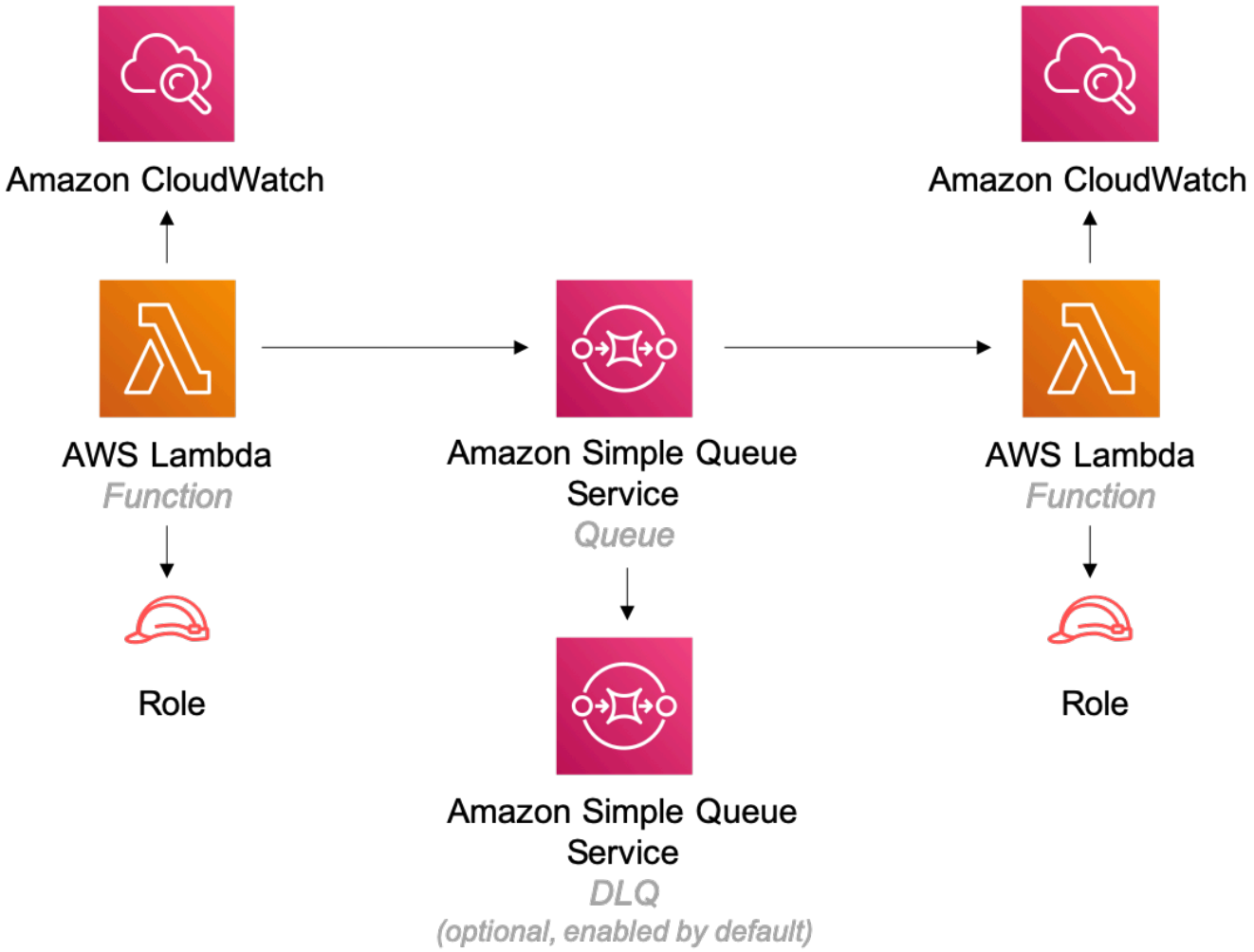
Funções do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para funções do Lambda
- Habilite a reutilização de conexões com Keep-Alive para funções do NodeJS Lambda.
- Habilitar rastreamento do X-Ray
- Definição de variáveis de ambiente:
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Fila do Amazon SQS

- Implantar uma dead letter queue para a fila principal.
- Habilitar a criptografia no lado do servidor para a fila principal usando uma chave do KMS gerenciada da AWS.
- Aplique a criptografia de dados em trânsito

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-lambda-sqs-lambda

aws-lambda-step-function

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos

àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_lambda_step_function`
TypeScript	`@aws-solutions-constructs/aws-lambda-step-function`
Java	`software.amazon.awsconstructs.services.lambdastepfunction`

Overview

Este AWS Solutions Construct implementa uma função do AWS Lambda conectada a uma função de etapa da AWS.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { LambdaToStepFunction } from '@aws-solutions-constructs/aws-lambda-step-
function';
import * as stepfunctions from '@aws-cdk/aws-stepfunctions';

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

new LambdaToStepFunction(this, 'LambdaToStepFunctionPattern', {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    },
```

```
    stateMachineProps: {
      definition: startState
    }
});
```

Initializer

```
new LambdaToStepFunction(scope: Construct, id: string, props:
 LambdaToStepFunctionProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`LambdaToStepFunctionProps`

Conceitos de criação de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo tanto isso e `lambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um `existingLambdaObj` é fornecido.
StateMachineProps	`sfn.StateMachineProps`	O usuário forneceu adereços para o SFN.StateMachine.

Nome	Tipo	Descrição
CreateCloudWatchAlms	`boolean`	Criar alarmes recomendados do CloudWatch.
LoggroupProps?	`logs.LogGroupProps`	Conceitos opcionais fornecido s pelo usuário para substitui r os adereços padrão para o grupo de logs do CloudWatch Logs.
StateMachineEnvironmentVari ableName	`string`	Nome opcional para o conjunto de variáveis de ambiente de máquina de estado Step Functions para a função Lambda produtor.

Propriedades de padrão

Nome	Tipo	Descrição
CloudwatchAlarm?	`cloudwatch.Alarm[]`	Retorna uma lista de um ou mais alarmes do CloudWatch criados pelo padrão.
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.
StateMachine	`sfn.StateMachine`	Retorna uma instância da máquina de estado criada pelo padrão.
StateMachineLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para a máquina de estado.

Configuração padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

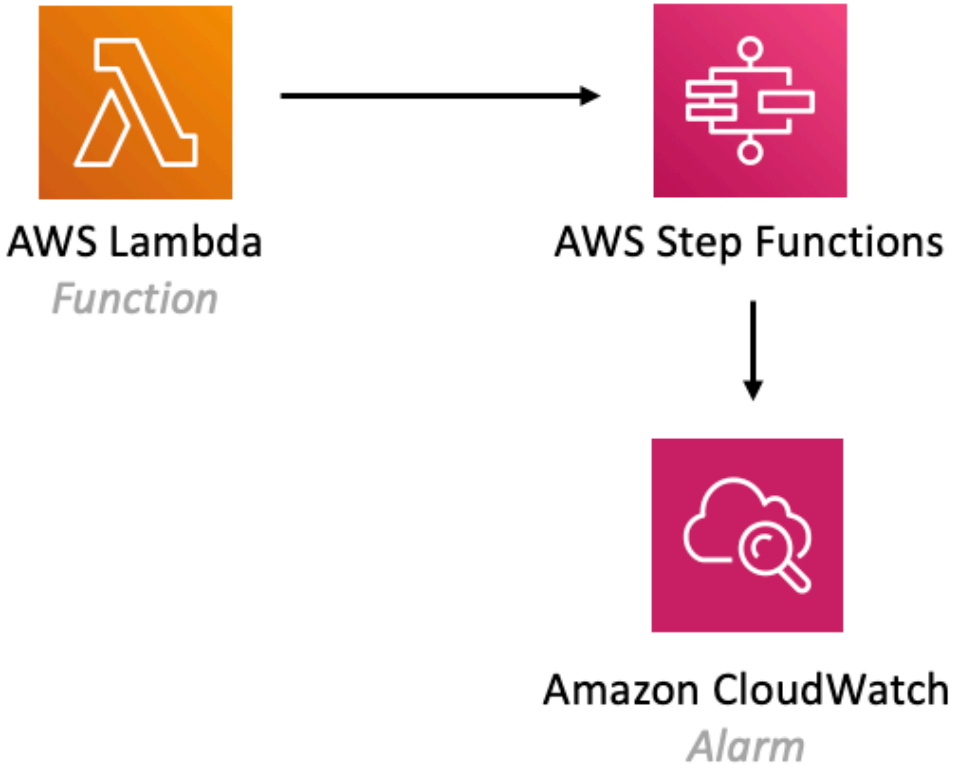
Função do AWS Lambda

- Configure uma função do IAM de acesso de privilégio limitado para a função do Lambda.
- Habilite a reutilização de conexões com Keep-Alive para funções do NodeJS Lambda.
- Ativar rastreamento do X-Ray.
- Definir variáveis de ambiente:
  - `STATE_MACHINE_ARN` (padrão)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Máquina de estado de funções de etapa da

- Implante alarmes de práticas recomendadas do CloudWatch para o AWS Step Functions State Machine

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-lambda-step-function

aws-s3-lambda

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_s3_lambda`
TypeScript	`@aws-solutions-constructs/aws-s3-lambda`
Java	`software.amazon.awsconstructs.services.s3lambda`

Overview

Este AWS Solutions Construct implementa um bucket do Amazon S3 conectado a uma função do AWS Lambda.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { S3ToLambdaProps, S3ToLambda } from '@aws-solutions-constructs/aws-s3-lambda';

new S3ToLambda(this, 'test-s3-lambda', {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    },
});
```

Initializer

```
new S3ToLambda(scope: Construct, id: string, props: S3ToLambdaProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`S3ToLambdaProps`

Adereços de construção de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo tanto isso `elambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.
ExistingBucketoBJ?	`s3.Bucket`	Instância existente do objeto S3 Bucket. Se isso for fornecido, então também fornecendo`bucketProps` é um erro.

Nome	Tipo	Descrição
Baldes?	`s3.BucketProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão do bucket. Ignorado se um`existingBucketObj` é fornecido.
S3EventSourceProps?	`S3EventSourceProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para S3EventSourceProps

Propriedades de padrão

Nome	Tipo	Descrição
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.
S3Bucket?	`s3.Bucket`	Retorna uma instância do bucket S3 criado pelo padrão.
S3loggingBucket?	`s3.Bucket`	Retorna uma instância do bucket de log criado pelo padrão para o bucket S3.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Amazon S3 Bucket

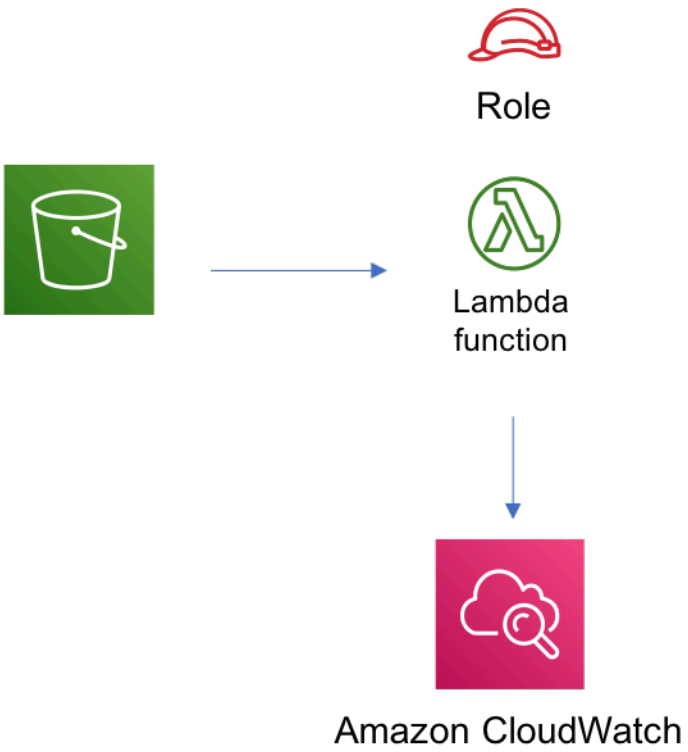
- Configurar o log de acesso para o S3 Bucket.

- Ative a criptografia do lado do servidor para o S3 Bucket usando a chave KMS gerenciada pela AWS.
- Ative o controle de versão para o bucket do S3.
- Não permitir acesso público para o S3 Bucket.
- Mantenha o bucket do S3 ao excluir a pilha do CloudFormation.
- Aplique a criptografia de dados em trânsito
- Aplica a regra de ciclo de vida para mover versões de objetos não atuais para o armazenamento do Glacier após 90 dias.

Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Ativar rastreamento de X-Ray
- Definição de variáveis de ambiente:
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-s3-lambda

aws-s3-sqs

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_s3_sqs`
TypeScript	`@aws-solutions-constructs/aws-s3-sqs`
Java	`software.amazon.awsconstructs.services.s3sqs`

Overview

Este AWS Solutions Construct implementa um bucket do Amazon S3 configurado para enviar notificações para uma fila do Amazon SQS.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { S3ToSqs } from "@aws-solutions-constructs/aws-s3-sqs";

new S3ToSqs(stack, 'S3ToSQSPattern', {});
```

Initializer

```
new S3ToSqs(scope: Construct, id: string, props: S3ToSqsProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`S3ToSqsProps`

Adereços de criação de padrão

Nome	Tipo	Descrição
ExistingBucketoBJ?	`s3.Bucket`	Instância existente do objeto S3 Bucket. Se isso for fornecido, então também fornecendo`bucketProps` é um erro.
Baldes?	`s3.BucketProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o bucket S3.
S3EventTypes?	`s3.EventType[]`	Os tipos de evento do S3 que acionarão a notificaç ão. Padronizado como

Nome	Tipo	Descrição
		`s3.EventType.OBJECT_CREATED` .
S3EventFilters?	`s3.NotificationKeyFilter[]`	As regras de filtro de chave de objeto S3 para determinar quais objetos acionam esse evento. Se não for especificado, nenhuma regra de filtro será aplicada.
ExistingQueueobj?	`sqs.Queue`	Uma fila SQS opcional existente a ser usada em vez da fila padrão. Fornecendo tanto isso quanto `queueProps` causará um erro. Se a fila SQS estiver criptografada, a chave KMS utilizada para criptografia deve ser um CMK gerenciado pelo cliente.
QueueProps?	`sqs.QueueProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da fila SQS. Ignorado se um `existingQueueObj` é fornecido.
DeadletterQueueProps?	`sqs.QueueProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para a fila de letras inativas. Usado somente se o `deployDeadLetterQueue` está definida como true.

Nome	Tipo	Descrição
Implementar Deadlette rQueue?	`boolean`	Criar uma fila secundária para ser usada como uma dead letter queue. Padronizado como `true`.
MaxReceiveCount?	`number`	O número de vezes que uma mensagem pode ser desenfile irada sem sucesso antes de ser movida para a fila de mensagens mortas. Padroniza do como 15.
EnableEncryptionWi thCustomerManagedKey?	`boolean`	Se deve usar uma Chave KMS, gerenciada por este aplicativo CDK ou importada. Se importar uma chave de criptogra fia, ela deve ser especific ada no campo`encryptio nKey` propriedade para esta construção.
encryptionKey?	`kms.Key`	Uma chave de criptografia opcional existente a ser usada em vez da chave de criptogra fia padrão.
EncryptionKeyProps?	`kms.KeyProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da chave de criptogra fia.

Propriedades do padrão

Nome	Tipo	Descrição
SQSqueue	`sqs.Queue`	Retorna uma instância da fila SQS criada pelo padrão.
DeadletterQueue?	`sqs.Queue`	Retorna uma instância da fila de letras mortas criada pelo padrão, se uma for implantad a.
encryptionKey	`kms.IKey`	Retorna uma instância da chave de criptografia criada pelo padrão.
S3Bucket?	`s3.Bucket`	Retorna uma instância do bucket S3 criado pelo padrão.
S3loggingBucket?	`s3.Bucket`	Retorna uma instância do bucket de log criado pelo padrão para o bucket S3.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

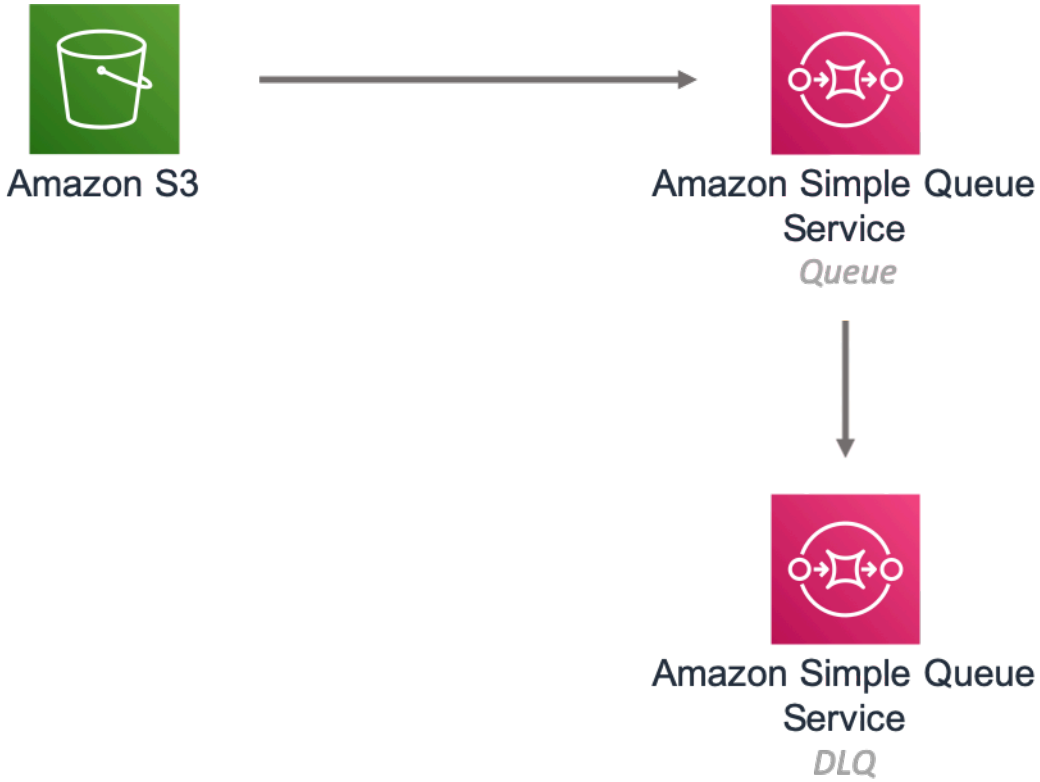
Amazon S3 Bucket

- Configurar registro de acesso para o bucket do S3
- Ativar criptografia no lado do servidor para o S3 Bucket usando a chave KMS gerenciada pela AWS
- Ativar o controle de versão para o bucket do S3
- Não permitir acesso público para o S3 Bucket
- Manter o bucket do S3 ao excluir a pilha do CloudFormation
- Aplique a criptografia de dados em trânsito
- Aplica regra de ciclo de vida para mover versões de objetos não atuais para o armazenamento do Glacier após 90 dias

Fila do Amazon SQS

- Configurar permissões de acesso de menor privilégio para SQS Queue
- Implantar fila de mensagens mortas do SQS para a fila do SQS de origem
- Habilitar a criptografia no lado do servidor para a fila do SQS usando a KMS Key
- Aplique a criptografia de dados em trânsito

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:

	@aws -solutions-constructs/aws-s3-sqs

aws-s3-step-function

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àControle de versão semânticaModelo Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_s3_step_function`
TypeScript	`@aws-solutions-constructs/aws-s3-step-function`
Java	`software.amazon.awsconstructs.services.s3stepfunction`

Overview

Este AWS Solutions Construct implementa um bucket do Amazon S3 conectado a uma função de etapa da AWS.

> **Note**
>
> Essa construção usa o Amazon EventBridge (Amazon CloudWatch Events) para acionar as AWS Step Functions. O EventBridge é mais flexível, mas acionar Step Functions com notificações de eventos do S3 tem menos latência e é mais econômico. Se o custo e/ou a latência forem um problema, você deve considerar a implantação`aws-s3-lambda`e`aws-lambda-stepfunctions`no lugar desta construção.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { S3ToStepFunction, S3ToStepFunctionProps } from '@aws-solutions-constructs/aws-s3-step-function';
import * as stepfunctions from '@aws-cdk/aws-stepfunctions';

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

new S3ToStepFunction(this, 'test-s3-step-function-stack', {
    stateMachineProps: {
      definition: startState
    }
});
```

Initializer

```
new S3ToStepFunction(scope: Construct, id: string, props: S3ToStepFunctionProps);
```

Parâmetros

- escopoConstruct
- idstring
- propsS3ToStepFunctionProps

Adereços de construção de padrão

Nome	Tipo	Descrição
ExistingBucketoBJ?	s3.IBucket	Instância existente do objeto S3 Bucket. Se isso for fornecido, então também fornecendobucketProps é um erro.
Baldes?	s3.BucketProps	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades

Nome	Tipo	Descrição
		padrão do bucket. Ignorado se um`existingBucketObj` é fornecido.
StateMachineProps	`sfn.StateMachineProps`	O usuário opcional forneceu adereços para substituir os adereços padrão para SFN.StateMachine.
Eventruleprops?	`events.RuleProps`	O usuário opcional forneceu EventruleProps para substituir os padrões.
Implementar CloudTrail?	`boolean`	Implantar uma trilha no AWS CloudTrail para registrar eventos de API no Amazon S3. Padronizado como `true`.
CreateCloudWatchAlms	`boolean`	Criar alarmes recomendados do CloudWatch.
LoggroupProps?	`logs.LogGroupProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para o grupo de logs do CloudWatch Logs.

Propriedades do padrão

Nome	Tipo	Descrição
Cloudtrail?	`cloudtrail.Trail`	Retorna uma instância da trilha Cloudtrail criada pelo padrão.
CloudTrailBucket?	`s3.Bucket`	Retorna uma instância do bucket criado pelo padrão

Nome	Tipo	Descrição
		para armazenar dados de trilha do Cloudtrail.
CloudTrailLoggingBucket?	`s3.Bucket`	Retorna uma instância do bucket de log criado pelo padrão para o bucket principal usado pela trilha do Cloudtrail.
CloudwatchAlarm?	`cloudwatch.Alarm[]`	Retorna uma lista de um ou mais alarmes do CloudWatch criados pelo padrão.
Balde?	`s3.Bucket`	Retorna uma instância do bucket S3 criado pelo padrão.
S3loggingBucket?	`s3.Bucket`	Retorna uma instância do bucket de log criado pelo padrão para o bucket S3.
StateMachine	`sfn.StateMachine`	Retorna uma instância da máquina de estado criada pelo padrão.
StateMachineLogGroup	`logs.LogGroup`	Retorna uma instância do grupo de logs criado pelo padrão para a máquina de estado.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

Bucket do Amazon S3

- Configurar o log de acesso para o S3 Bucket.
- Ative a criptografia do lado do servidor para o S3 Bucket usando a chave KMS gerenciada pela AWS.

- Ative o controle de versão para o bucket do S3.
- Não permitir acesso público para o S3 Bucket.
- Mantenha o bucket do S3 ao excluir a pilha do CloudFormation.
- Aplique a criptografia de dados em trânsito
- Aplica a regra de ciclo de vida para mover versões de objetos não atuais para o armazenamento do Glacier após 90 dias.

AWS CloudTrail

- Configure uma trilha no AWS CloudTrail para registrar eventos de API no Amazon S3 relacionados ao bucket criado pelo Construct.

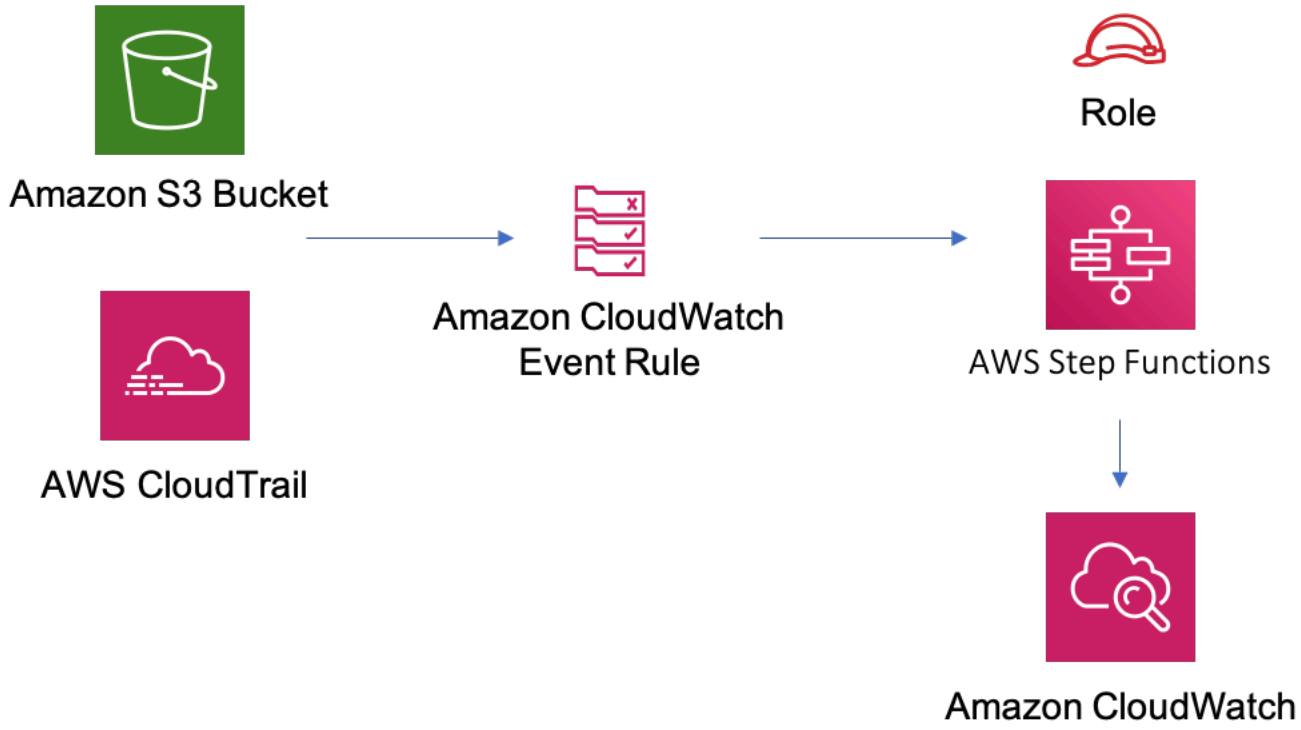
Amazon CloudWatch Events Regra

- Conceda permissões de menor privilégio ao CloudWatch Events para acionar a Função Lambda.

AWS Step Function

- Ative o log do CloudWatch para API Gateway
- Implante as práticas recomendadas Alarmes do CloudWatch para a função Step.

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-s3-step-function

aws-sns-lambda

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àSemantic versionModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_sns_lambda`
TypeScript	`@aws-solutions-constructs/aws-sns-lambda`
Java	`software.amazon.awsconstructs.services.snslambda`

Overview

Este AWS Solutions Construct implementa um Amazon SNS conectado a uma função do AWS Lambda.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { SnsToLambda, SnsToLambdaProps } from "@aws-solutions-constructs/aws-sns-
lambda";

new SnsToLambda(this, 'test-sns-lambda', {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    }
});
```

Initializer

```
new SnsToLambda(scope: Construct, id: string, props: SnsToLambdaProps);
```

Parâmetros

- escopoConstruct
- idstring
- propsSnsToLambdaProps

Adereços de criação de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	lambda.Function	Instância existente do objeto Lambda Function, fornecendo tanto isso elambdaFunctionProps causará um erro.
LambdaFunctionProps?	lambda.FunctionProps	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da função Lambda. Ignorado se umexistingLambdaObj é fornecido.
ExistingTopicobj?	sns.Topic	Instância existente do objeto Tópico SNS, fornecendo tanto isso quantotopicProps causará um erro.
TopicProps?	sns.TopicProps	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão do tópico SNS.

Propriedades de padrão

Nome	Tipo	Descrição
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.
snsTopic	`sns.Topic`	Retorna uma instância do tópico SNS criado pelo padrão.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

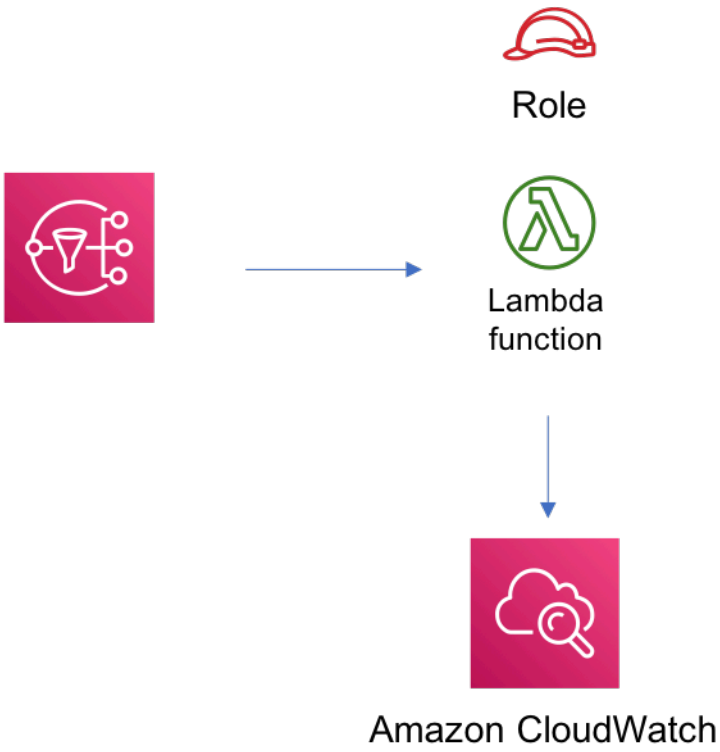
Tópico do Amazon SNS

- Configurar permissões de acesso de menor privilégio para o tópico SNS.
- Habilite a criptografia do lado do servidor usando a chave KMS gerenciada pela AWS.
- Aplique a criptografia de dados em trânsito

Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Ative rastreamento de X-Ray.
- Configuração de variáveis de ambiente:
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-sns-lambda

aws-sns-sqs

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àControle de versões semânticasModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_sns_sqs`
TypeScript	`@aws-solutions-constructs/aws-sns-sqs`
Java	`software.amazon.awsconstructs.services.snssqs`

Overview

Este AWS Solutions Construct implementa um tópico do Amazon SNS conectado a uma fila do Amazon SQS.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
import { SnsToSqs, SnsToSqsProps } from "@aws-solutions-constructs/aws-sns-sqs";
import * as iam from '@aws-cdk/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);
```

Initializer

```
new SnsToSqs(scope: Construct, id: string, props: SnsToSqsProps);
```

Parâmetros

- escopoConstruct
- idstring
- propsSnsToSqsProps

Adereços de construção de padrão

Nome	Tipo	Descrição
ExistingTopicobj?	sns.Topic	Instância existente do objeto Tópico SNS, fornecendo tanto isso quantotopicProps causará um erro.
TopicProps?	sns.TopicProps	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão do tópico SNS. Ignorado se umexistingTopicObj é fornecido.
ExistingQueueobj?	sqs.Queue	Uma fila SQS opcional existente a ser usada em vez da fila padrão. Fornecendo tanto isso quantoqueueProps causará um erro.
QueueProps?	sqs.QueueProps	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades

Nome	Tipo	Descrição
		padrão da fila SQS. Ignorado se um`existingQueueObj` é fornecido.
Implementar DeadletterQueue?	`boolean`	Se criar uma fila secundária para ser usada como uma dead letter queue Padronizado como `true`.
DeadletterQueueProps?	`sqs.QueueProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para a fila de letras inativas. Usado somente se o`deployDeadLetterQueue` está definida como true.
MaxReceiveCount?	`number`	O número de vezes que uma mensagem pode ser desenfileirada sem sucesso antes de ser movida para a dead letter queue. Padronizado como `15`.
EnableEncryptionWithCustomerManagedKey?	`boolean`	Se deve usar uma chave de criptografia gerenciada pelo cliente, gerenciada por este aplicativo CDK ou importada. Se importar uma chave de criptografia, ela deve ser especificada na caixa`encryptionKey` propriedade para esta construção.

Nome	Tipo	Descrição
encryptionKey	`kms.Key`	Uma chave de criptografia opcional existente a ser usada em vez da chave de criptogra fia padrão.
EncryptionKeyProps?	`kms.KeyProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da chave de criptogra fia.

Propriedades de padrão

Nome	Tipo	Descrição
snsTopic	`sns.Topic`	Retorna uma instância do tópico SNS criado pelo padrão.
encryptionKey	`kms.Key`	Retorna uma instância da chave de criptografia criada pelo padrão.
SQSqueue	`sqs.Queue`	Retorna uma instância da fila SQS criada pelo padrão.
DeadletterQueue?	`sqs.Queue`	Retorna uma instância da fila de letras mortas criada pelo padrão, se uma for implantad a.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

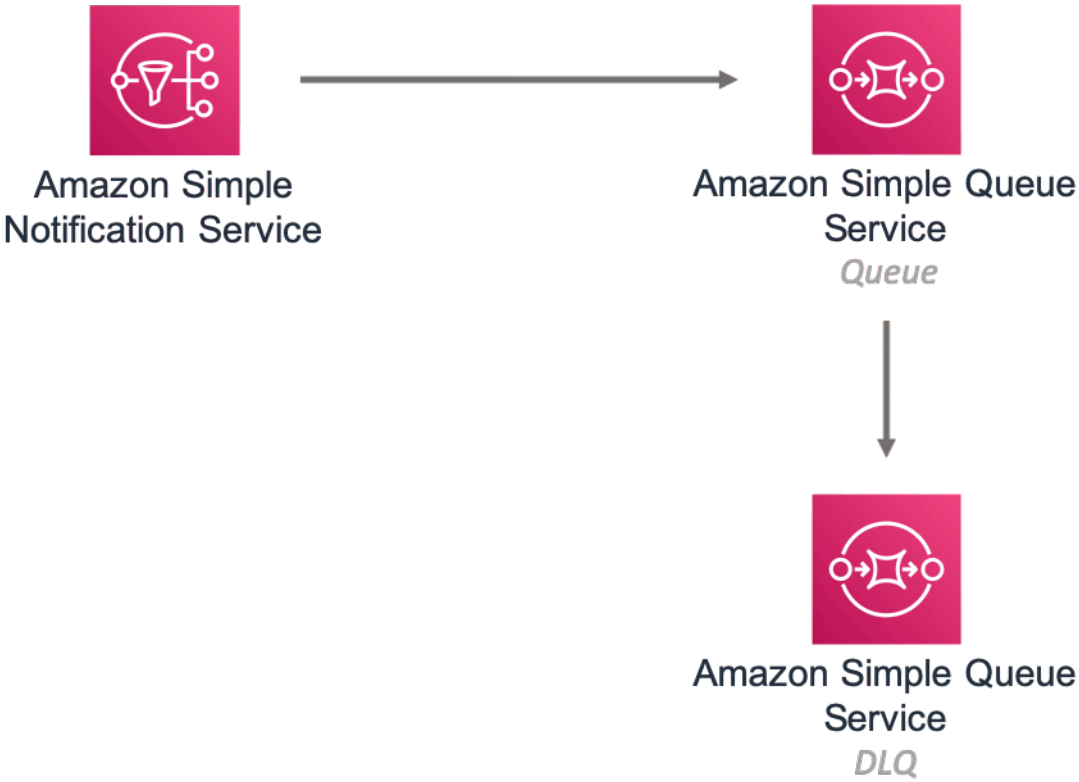
Tópico do Amazon SNS

- Configurar permissões de acesso de menor privilégio para o tópico SNS.
- Habilite a criptografia do lado do servidor usando a chave KMS gerenciada pela AWS.
- Aplique a criptografia de dados em trânsito

Fila do Amazon SQS

- Configure as permissões de acesso de menor privilégio para a fila SQS.
- Implante a fila de mensagens mortas para a fila do SQS de origem.
- Habilitar a criptografia no lado do servidor para a fila do SQS usando a chave do KMS gerenciada pelo cliente.
- Aplique a criptografia de dados em trânsito

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-sns-sqs

aws-sqs-lambda

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àControle de versão semânticaModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

Observações: Para garantir a funcionalidade adequada, os pacotes AWS Solutions Constructs e os pacotes CDK da AWS em seu projeto devem ser da mesma versão.

Linguagem	Pacote
Python	`aws_solutions_constructs.aws_sqs_lambda`
TypeScript	`@aws-solutions-constructs/aws-sqs-lambda`
Java	`software.amazon.awsconstructs.services.sqslambda`

Overview

Este AWS Solutions Construct implementa uma fila do Amazon SQS conectada a uma função do AWS Lambda.

Aqui está uma definição de padrão implantável mínima no TypeScript:

```
const { SqsToLambda } = require('@aws-solutions-constructs/aws-sqs-lambda');

new SqsToLambda(stack, 'SqsToLambdaPattern', {
    lambdaFunctionProps: {
        runtime: lambda.Runtime.NODEJS_14_X,
        // This assumes a handler function in lib/lambda/index.js
        code: lambda.Code.fromAsset(`${__dirname}/lambda`),
        handler: 'index.handler'
    }
});
```

Initializer

```
new SqsToLambda(scope: Construct, id: string, props: SqsToLambdaProps);
```

Parâmetros

- escopo`Construct`
- id`string`
- props`SqsToLambdaProps`

Adereços de construção de padrão

Nome	Tipo	Descrição
ExistingAmbdaobj?	`lambda.Function`	Instância existente do objeto Lambda Function, fornecendo tanto isso `elambdaFunctionProps` causará um erro.
LambdaFunctionProps?	`lambda.FunctionProps`	Propriedades opcionais fornecidas pelo usuário para

Nome	Tipo	Descrição
		substituir as propriedades padrão da função Lambda. Ignorado se um`existingLambdaObj` é fornecido.
ExistingQueueobj?	`sqs.Queue`	Uma fila SQS opcional existente a ser usada em vez da fila padrão. Fornecendo tanto isso quanto`queueProps` causará um erro.
QueueProps?	`sqs.QueueProps`	Propriedades opcionais fornecidas pelo usuário para substituir as propriedades padrão da fila SQS. Ignorado se um`existingQueueObj` é fornecido.
ImplantyDeadletterQueue?	`boolean`	Criar uma fila secundária para ser usada como uma dead letter queue. Padronizado como `true`.
DeadletterQueueProps?	`sqs.QueueProps`	Props opcionais fornecidos pelo usuário para substituir os adereços padrão para a fila de letras inativas. Usado somente se o`deployDeadLetterQueue` está definida como true.
MaxReceiveCount?	`number`	O número de vezes que uma mensagem pode ser desenfileirada sem sucesso antes de ser movida para a fila de mensagens mortas. Padronizado como 15.

Propriedades de padrão

Nome	Tipo	Descrição
DeadletterQueue?	`sqs.Queue`	Retorna uma instância da fila de letras mortas criada pelo padrão, se uma for implantad a.
LambdaFunction	`lambda.Function`	Retorna uma instância da função Lambda criada pelo padrão.
SQSqueue	`sqs.Queue`	Retorna uma instância da fila SQS criada pelo padrão.

Configurações padrão

A implementação imediata desse padrão sem substituições definirá os seguintes padrões:

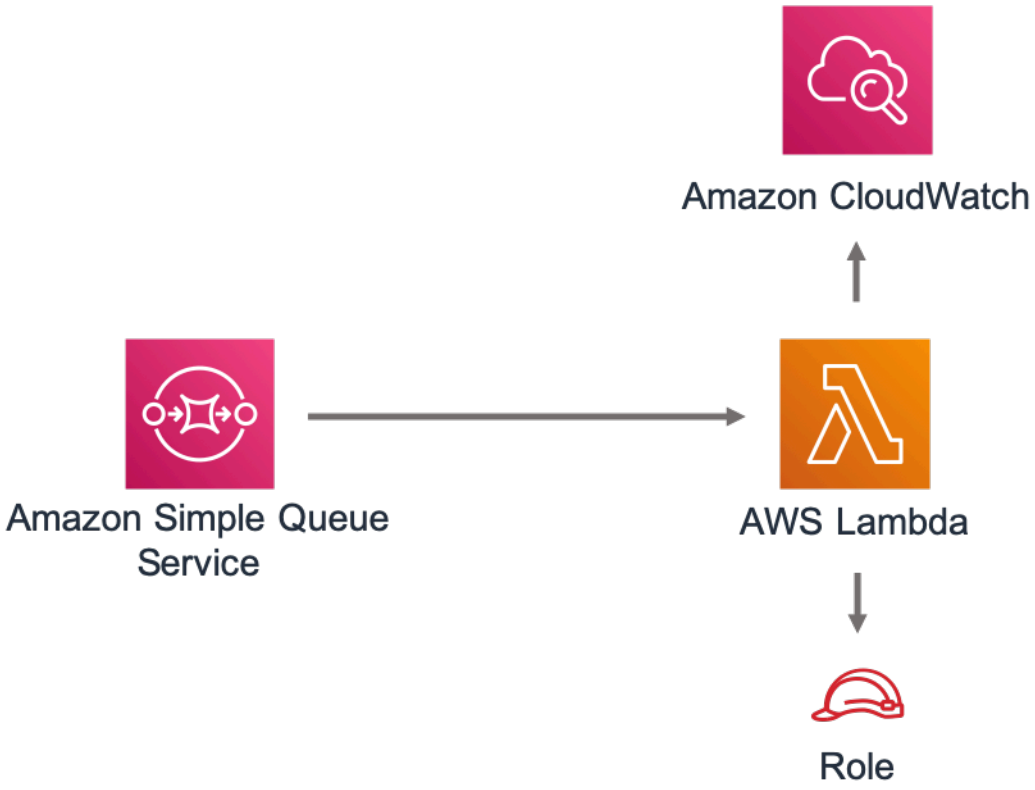
Fila do Amazon SQS

- Implantar fila de mensagens mortas do SQS para a fila do SQS de origem.
- Habilitar a criptografia no lado do servidor para a fila do SQS de origem usando a AWS Managed KMS Key.
- Aplique a criptografia de dados em trânsito

Função do AWS Lambda

- Configurar a função do IAM de acesso de privilégio limitado para a função Lambda
- Habilite a reutilização de conexões com a função Keep-Alive para NodeJS Lambda.
- Habilitar rastreamento do X-Ray
- Definição de variáveis de ambiente:
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(para funções Node 10.x e superiores)

Architecture



GitHub

Para exibir o código desse padrão, crie/exiba problemas e solicitações pull e muito mais:	
	@aws -solutions-constructs/aws-sqs-lambda

core

STABILITY EXPERIMENTAL

Todas as classes estão em desenvolvimento ativo e estão sujeitas a alterações ou remoção não compatíveis com versões anteriores em qualquer versão futura. Estes não estão sujeitos àVersionamento semânticoModelo. Isso significa que, embora você possa usá-los, você pode precisar atualizar seu código-fonte ao atualizar para uma versão mais recente deste pacote.

A biblioteca principal inclui os blocos de construção básicos dos AWS Solutions Constructs. Ele define as classes principais que são usadas no resto dos AWS Solutions Constructs.

Propriedades padrão para construções CDK da AWS

A biblioteca central define as propriedades padrão para as construções CDK da AWS usadas por construções de AWS Solutions Constructs.

Por exemplo, a seguir está o trecho de propriedades padrão para o construtor do S3 Bucket criado pelo construtor do AWS Solutions Constructs. Por padrão, ele ativará a criptografia do lado do servidor, o controle de versão do bucket, bloqueará todo o acesso público e configurará o log de acesso do S3.

```
{
  encryption: s3.BucketEncryption.S3_MANAGED,
  versioned: true,
  blockPublicAccess: s3.BlockPublicAccess.BLOCK_ALL,
  removalPolicy: RemovalPolicy.RETAIN,
  serverAccessLogsBucket: loggingBucket
}
```

Substituir as propriedades padrão

As propriedades padrão definidas pela biblioteca Core podem ser substituídas pelas propriedades fornecidas pelo usuário. Por exemplo, o usuário pode substituir a propriedade Block Public Access do Amazon S3 para atender a requisitos específicos.

```
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
  },
});
```

Avisos de substituição de propriedades

Quando uma propriedade padrão da biblioteca Core é substituída por uma propriedade fornecida pelo usuário, Constructs emitirá uma ou mais mensagens de aviso para o console destacando a (s) alteração (ões). Essas mensagens são destinadas a fornecer reconhecimento situacional ao usuário e evitar substituições não intencionais que possam criar riscos à segurança. Essas mensagens aparecerão sempre que comandos relacionados à implantação/compilação forem executados, incluindo`cdk deploy`,`cdk synth`,`npm test`, etc.

Exemplos de mensagem:`AWS_CONSTRUCTS_WARNING: An override has been provided for the property: BillingMode. Default value: 'PAY_PER_REQUEST'. You provided: 'PROVISIONED'.`

Alternar avisos de substituição

As mensagens de aviso de substituição são ativadas por padrão, mas podem ser ativadas/desativadas explicitamente usando o comando`overrideWarningsEnabled`Variável shell.

- Para explicitamenteDesativar osubstituir avisos, executar`export overrideWarningsEnabled=false`.
- Para explicitamenteAtivar osubstituir avisos, executar`export overrideWarningsEnabled=true`.
- Para reverter para o padrão dou`unset overrideWarningsEnabled`.

Revisões do documento

Para ser notificado sobre atualizações do AWS Solutions Constructs, inscreva-se no feed RSS.

update-history-change	update-history-description	update-history-date
Conteúdo atualizado	Adicionado padrão aws-lambda-ssmstringparameter. Outras atualizações menores de conteúdo.	27 de maio de 2021
Conteúdo atualizado	Adicionado padrão aws-lambda-secretsmanager. Outras atualizações menores de conteúdo.	12 de maio de 2021
Conteúdo atualizado	Atualizações de propriedade para selecionar padrões*-lambda. Outras atualizações menores de conteúdo.	17 de abril de 2021
Conteúdo atualizado	Corrigido um problema no Passo a passo para usuários Python e exemplos de propriedades atualizados para construções contendo funções do Lambda.	30 de março de 2021
Conteúdo atualizado	Pequenas correções/atualizações para adereços padrão e configurações padrão para padrões selecionados.	8 de março de 2021
Conteúdo atualizado	Pequenas correções/atualizações para conteúdo passo a passo.	4 de março de 2021

Conteúdo atualizado	Adição do`aws-lambda-sagemakerendpoint` e propriedades atualizadas para determinados padrões do Kinesis Firehose.	24 de fevereiro de 2021
Conteúdo atualizado	Adição do`aws-kinesisstreams-gluejob`padrão e etapas passo a passo atualizadas para usuários Python.	17 de fevereiro de 2021
Conteúdo atualizado	Propriedades atualizadas para`aws-cloudfront-*`Padrões.	9 de fevereiro de 2021
Conteúdo atualizado	Adicionado link para o GitHub para cada padrão.	5 de fevereiro de 2021
Conteúdo atualizado	Propriedades atualizadas para padrões selecionados.	1º de fevereiro de 2021
Conteúdo atualizado	Documentação atualizada de propriedades e configurações padrão para padrões selecionados.	4 de janeiro de 2021
Conteúdo atualizado	Adicionado novos padrões: aws-cloudfront-mediastore e aws-s3-sqs.	20 de dezembro de 2020
Conteúdo atualizado	Removido padrão aws-lambda-sagemaker.	17 de novembro de 2020

Conteúdo atualizado	Adicionado novos padrões: aws-events-rule-kinesisstreams, aws-events-rule-kinesisfirehose-s3 e aws-lambda-sagemaker.	27 de outubro de 2020
Conteúdo atualizado	Atualizado para refletir a mudança de quebra nos padrões aws-events-rule-sns e aws-events-rule-sqs: nomes de classe e interface alterados para caso pascal.	22 de outubro de 2020
Conteúdo atualizado	Adicionado aws-apigateway-sagemakerendpoint e aws-kinesisstreams-kinesisfirehose-s3 padrões; outras pequenas atualizações para o conteúdo existente.	20 de outubro de 2020
Conteúdo atualizado	Adicionado padrão aws-apigateway-iot; outras pequenas atualizações para o conteúdo existente.	7 de outubro de 2020
Conteúdo atualizado	Atualizado snippets mínimos de código de padrão implantáveis e padrões de práticas recomendadas para todos os padrões.	5 de outubro de 2020
Conteúdo atualizado	Propriedades atualizadas para o padrão aws-kinesisstreams-lambda para refletir mudanças de quebra.	14 de setembro de 2020

Conteúdo atualizado	Correção menor para a segunda parte do passo a passo.	10 de setembro de 2020
Conteúdo atualizado	Adicionado aws-apigateway-kinesisstreams, aws-events-rule-sns e aws-events-rule-sqs padrões.	10 de setembro de 2020
Conteúdo atualizado	Adicionado padrão aws-sns-sqs; atualizações para todos os padrões de SNS; pequenas correções tipográficas.	2 de setembro de 2020
Conteúdo atualizado	Corrigidos nomes de módulos para o padrão aws-sqs-l ambda.	31 de agosto de 2020
Conteúdo atualizado	Corrigido o nome do módulo Python para o padrão aws-dynamodb-stream-lambda-elasticsearch-kibana.	31 de agosto de 2020
Conteúdo atualizado	Padrão atualizado para padrões do Lambda; outras atualizações menores.	27 de agosto de 2020
Conteúdo atualizado	Propriedades públicas atualizadas para padrões do S3; padrões atualizados para padrões do DynamoDB.	10 de agosto de 2020
Conteúdo atualizado	Vários padrões atualizados para destacar a imposição padrão da criptografia em trânsito.	4 de agosto de 2020

Conteúdo atualizado	Adicionado padrão aws-lambda-sqs-lambda; instruções de configuração aprimoradas no Guia de Introdução; atualizado todos os padrões para disponibilizar recursos adicionais por meio de propriedades públicas.	27 de julho de 2020
Conteúdo atualizado	Adicionado padrão aws-lambda-sqs; outras atualizações menores.	20 de julho de 2020
Conteúdo atualizado	As propriedades DeployLambda e DeployBucket foram removidas de padrões relevantes; outras atualizações menores.	9 de julho de 2020
Conteúdo atualizado	Adicionado padrão aws-lambda-step-function e corrigido erros tipográficos menores.	7 de julho de 2020
Conteúdo atualizado	Adição do TableObj? para selecionar padrões do DynamoDB.	25 de junho de 2020
Conteúdo atualizado	Várias correções de texto e correções para links quebrados.	23 de junho de 2020
Versão inicial	Construtos de soluções da AWS disponibilizados publicamente.	22 de junho de 2020

Notices

Os clientes são responsáveis por fazer sua própria avaliação independente das informações contidas neste documento. Este documento: (a) é apenas para fins informativos, (b) representa as ofertas e práticas atuais de produtos da AWS, que estão sujeitas a alterações sem aviso prévio, e (c) não cria quaisquer compromissos ou garantias da AWS e de suas afiliadas, fornecedores ou licenciantes. Os produtos ou serviços da AWS são fornecidos “tal como estão” sem garantias, representações ou condições de qualquer tipo, expressas ou implícitas. As responsabilidades e as obrigações da AWS com os seus clientes são controladas por contratos da AWS, e este documento não é parte, nem modifica, qualquer contrato entre a AWS e seus clientes.

© 2020 Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.