Solutions AWS

Constructions dans les Solutions d'AWS

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

Constructions dans les Solutions d'AWS: Solutions AWS

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

Les marques commerciales et la présentation commerciale d'Amazon ne peuvent pas être utilisées en relation avec un produit ou un service extérieur à Amazon, d'une manière susceptible d'entraîner une confusion chez les clients, ou d'une manière qui dénigre ou discrédite Amazon. Toutes les autres marques commerciales qui ne sont pas la propriété d'Amazon sont la propriété de leurs propriétaires respectifs, qui peuvent ou non être affiliés ou connectés à Amazon, ou sponsorisés par Amazon.

Table of Contents

Présentation ... 1
Présentation d'AWS Solutions ... 1
Pourquoi utiliser AWS Solutions Constructs ? ... 1
Mise en route ... 3
Prérequis ... 3
Installation du CDK AWS ... 4
Utilisation des constructions AWS Solutions ... 4
Procédure pas à pas - Partie 1 ... 5
Hello Constructions ... 5
Création de l'annuaire des applications et initialisation du CDK AWS ... 6
Modification des dépendances de base de projet ... 7
Code de gestionnaire Lambda ... 9
Installer les dépendances AWS CDK et AWS Solutions Construit ... 10
Ajouter un modèle Amazon API Gateway/AWS Lambda à votre pile ... 12
Déploiement sur cdk ... 18
Sortie de pile ... 19
Tester votre application ... 19
Procédure pas à pas - Partie 2 ... 19
Code de compteur Lambda ... 20
Installer les nouvelles dépendances ... 22
Définissez les ressources ... 23
Vérifiez les modifications ... 36
Déploiement CDK ... 37
Sortie de pile ... 38
Tester votre application ... 38
Exemple de cas d'utilisation ... 39
Site Web AWS Static S3 ... 40
Gestionnaire d'images simple sans serveur AWS ... 40
AWS plémentation web sans serveur ... 40
Référence API ... 42
Modules ... 42
Contenu du module ... 42
aws-apigateway-dynamodb ... 43
Présentation ... 43

Initialiser ... 44
Modèle Construire des accessoires ... 44
Propriétés du modèle ... 46
Paramètres par défaut ... 46
Architecture ... 47
GitHub ... 48
aws-apigateway-iot ... 48
Présentation ... 49
Initialiseur ... 49
Modèle de construction des accessoires ... 50
Propriétés du modèle ... 51
Paramètres par défaut ... 51
Architecture ... 55
Exemples ... 55
GitHub ... 57
aws-apigateway-kinesisstreams ... 57
Présentation ... 58
Initialiseur ... 58
Accessoires de construction de modèle ... 59
Propriétés de modèle ... 60
Exemple d'utilisation de l'API ... 61
Paramètres par défaut ... 62
Architecture ... 63
GitHub ... 63
aws-apigateway-lambda ... 63
Présentation ... 64
Initialisation ... 65
Accessoires de construction de modèle ... 65
Propriétés de modèle ... 66
Paramètres par défaut ... 66
Architecture ... 67
GitHub ... 68
aws-apigateway-sagemakerendpoint ... 68
Présentation ... 69
Initialiseur ... 69
Accessoires de construction de modèle ... 70

Propriétés du modèle 71
Exemple d'utilisation de l'API 61
Paramètres par défaut 72
Architecture 73
GitHub 73
aws-apigateway-sqs 73
Présentation 74
Initialisation 74
Accessoires de construction de modèle 75
Propriétés du modèle 77
Exemples de scénario d'utilisation 61
Paramètres par défaut 78
Architecture 79
GitHub 80
aws-cloudfront-apigateway 80
Présentation 81
Initialiseur 81
Accessoires de construction 82
Propriétés de modèle 83
Paramètres par défaut 83
Architecture 84
GitHub 84
aws-cloudfront-apigateway-lambda 84
Présentation 85
Initialiseur 86
Accessoires de construction de modèle 86
Propriétés du modèle 87
Paramètres par défaut 88
Architecture 89
GitHub 89
aws-cloudfront-mediastore 90
Présentation 90
Initialiseur 91
Accessoires de construction de modèle 91
Propriétés du modèle 92
Paramètres par défaut 93

Architecture 94
GitHub 94
aws-cloudfront-s3 94
Présentation 95
Initialisation 95
Accessoires de construction de modèle 96
Propriétés du modèle 97
Paramètres par défaut 97
Architecture 98
GitHub 99
aws-cognito-apigateway-lambda 99
Présentation 81
Initialiseur 101
Props de construction de modèle 101
Propriétés de modèle 102
Paramètres par défaut 103
Architecture 105
GitHub 105
aws-dynamodb-stream-lambda 106
Présentation 106
Initialisation 107
Modèle de construction d'accessoires 107
Propriétés du modèle 108
Fonction Lambda 108
Paramètres par défaut 109
Architecture 110
GitHub 110
aws-dynamodb-stream-lambda-elasticsearch-kibana 110
Présentation 111
Initialiseur 112
Modèle de construction de modèle 112
Propriétés du modèle 113
Fonction Lambda 114
Paramètres par défaut 114
Architecture 116
GitHub 116

aws-evenements-rule-kinesisfirehose-s3 116
Présentation 117
Initialisation 118
Accessoires de construction de modèle 118
Propriétés de modèle 119
Paramètres par défaut 120
Architecture 121
GitHub 121
aws-events-rule-kinesisstreams 121
Présentation 122
Initialiseur 123
Modèle de construction d'accessoires 123
Propriétés de modèle 124
Paramètres par défaut 124
Architecture 125
GitHub 125
aws-événements-rule-lambda 125
Présentation 126
Initialiseur 127
Modèle 127
Propriétés de modèle 128
Paramètres par défaut 128
Architecture 129
GitHub 129
aws-événements-rule-sns 129
Présentation 130
Initialiseur 131
Modèle de construction d'accessoires 131
Propriétés du modèle 132
Paramètres par défaut 133
Architecture 134
GitHub 134
aws-evenements-rule-sqs 134
Présentation 135
Initialiseur 136
Modèle de construction 136

Propriétés de modèle 138
Paramètres par défaut 139
Architecture 140
GitHub 140
aws-events-rule-step-function 140
Présentation 141
Initialiseur 142
Accessoires de construction de modèle 142
Propriétés de modèle 143
Paramètres par défaut 143
Architecture 144
GitHub 144
aws-iot-kinesisfirehose-s3 144
Présentation 145
Initialiseur 146
Accessoires de construction de modèle 146
Propriétés du modèle 147
Paramètres par défaut 148
Architecture 149
GitHub 150
aws-iot-lambda 150
Présentation 151
Initialiseur 151
Accessoires de construction de modèle 152
Propriétés de modèle 152
Paramètres par défaut 153
Architecture 153
GitHub 154
aws-iot-lambda-dynamodb 154
Présentation 155
Initialisation 155
Modèle de construction 156
Propriétés de modèle 157
Paramètres par défaut 157
Architecture 158
GitHub 158

aws-kinesisfirehose-s3 159
Présentation 159
Initialisation 160
Accessoires de construction de modèle 160
Propriétés du modèle 161
Paramètres par défaut 162
Architecture 163
GitHub 163
aws-kinesisfirehose-s3-et-kinesisanalytics 163
Présentation 164
Initialiseur 165
Accessoires de construction de modèle 166
Propriétés du modèle 167
Paramètres par défaut 167
Architecture 169
GitHub 169
aws-kinesisstreams-gluejob 169
Présentation 170
Initialiseur 171
Accessoires de construction de modèle 172
SinkDatastoreProps 173
SinkStoreType 174
Paramètres par défaut 174
Architecture 176
GitHub 176
aws-kinesisstreams-kinesisfirehose-s3 176
Présentation 177
Initialiseur 177
Accessoires de construction de modèle 178
Propriétés du modèle 179
Paramètres par défaut 180
Architecture 181
GitHub 182
aws-kinesisstreams-lambda 182
Présentation 183
Initialiseur 183

Accessoires de construction de modèle 184
Propriétés de modèle 185
Paramètres par défaut 185
Architecture 186
GitHub 186
aws-lambda-dynamodb 186
Présentation 187
Initialiseur 188
Accessoires de construction de modèle 188
Propriétés du modèle 191
Paramètres par défaut 192
Architecture 193
GitHub 193
aws-lambda-elasticsearch-kibana 193
Présentation 194
Initialiseur 195
Accessoires de construction de modèle 195
Propriétés du modèle 196
Fonction Lambda 197
Paramètres par défaut 197
Architecture 199
GitHub 199
aws-lambda-s3 200
Présentation 200
Initialiseur 201
Accessoires de construction de modèle 201
Propriétés du modèle 205
Paramètres par défaut 206
Architecture 207
GitHub 207
aws-lambda-ssmstringparameter 207
Présentation 208
Initialiseur 209
Accessoires de construction de modèle 209
Propriétés de modèle 213
Paramètres par défaut 213

Architecture 214
GitHub 214
aws-lambda-sagemakerendpoint 214
Présentation 215
Initialiseur 216
Modèle de construction 216
Propriétés de modèle 221
Paramètres par défaut 221
Architecture 222
GitHub 223
aws-lambda-secretsmanager 223
Présentation 224
Initialiseur 224
Accessoires de construction de modèle 225
Propriétés du modèle 227
Paramètres par défaut 228
Architecture 229
GitHub 229
aws-lambda-sns 229
Présentation 230
Initialiseur 231
Accessoires de construction de modèle 231
Propriétés du modèle 234
Paramètres par défaut 235
Architecture 236
GitHub 236
aws-lambda-sqs 236
Présentation 237
Initialiseur 237
Accessoires de construction de modèle 238
Propriétés du modèle 242
Paramètres par défaut 243
Architecture 244
GitHub 244
aws-lambda-sqs-lambda 245
Présentation 245

Initialiseur 246
Modèle de construction 246
Propriétés du modèle 249
Paramètres par défaut 249
Architecture 250
GitHub 251
aws-lambda-step-function 251
Présentation 252
Initialiseur 252
Accessoires de construction de modèle 253
Propriétés de modèle 254
Paramètres par défaut 254
Architecture 255
GitHub 255
aws-s3-lambda 256
Présentation 256
Initialisation 257
Accessoires de construction de modèle 257
Propriétés du modèle 258
Paramètres par défaut 259
Architecture 260
GitHub 260
aws-s3-sqs 260
Présentation 261
Initialiseur 261
Accessoires de construction de modèle 262
Propriétés du modèle 264
Paramètres par défaut 265
Architecture 266
GitHub 266
aws-s3-step-function 266
Présentation 267
Initialiseur 268
Accessoires de construction de modèle 268
Propriétés du modèle 269
Paramètres par défaut 270

Architecture 272
GitHub 272
aws-sns-lambda 272
Présentation 273
Initialiseur 273
Accessoires de construction de modèle 274
Propriétés de modèle 275
Paramètres par défaut 275
Architecture 276
GitHub 276
aws-sns-sqs 276
Présentation 277
Initialiseur 278
Accessoires de construction de modèle 278
Propriétés de modèle 280
Paramètres par défaut 281
Architecture 281
GitHub 282
aws-sqs-lambda 282
Présentation 283
Initialiseur 283
Accessoires de construction de modèle 283
Propriétés de modèle 285
Paramètres par défaut 286
Architecture 287
GitHub 287
principal 287
Propriétés par défaut pour les constructions CDK AWS 288
Remplacer les propriétés par défaut 288
Avertissements de propriété 289
Révisions du document 290
Avis 295
..... ccxcvi

Constructions de solutions AWS

Date de publication : Mai 2021(Révisions du document)

Présentation d'AWS Solutions

AWS Solutions Constructs (Constructs) est une extension open source deCloud Development Kit (AWS CDK)qui fournit des modèles multi-services et bien architectés pour définir rapidement des solutions dans le code afin de créer une infrastructure prévisible et reproductible. L'objectif est d'accélérer l'expérience des développeurs pour créer des solutions de toute taille à l'aide de définitions basées sur des motifs pour leur architecture.

Utilisez AWS Solutions Constructs pour définir vos solutions dans un langage de programmation familier. AWS Solutions Constructs prend en charge TypeScript, JavaScript, Python et Java pour le moment.

Pour parcourir le catalogue complet des modèles AWS Solutions Constructs,Cliquez ici.

Pourquoi utiliser AWS Solutions Constructs ?

Avec le rythme d'innovation des fournisseurs de cloud, connaître et comprendre les meilleures pratiques et s'assurer qu'elles sont correctement mises en œuvre dans l'ensemble de votre solution peut être intimidant. Constructs vous permet de combiner des modèles prédéfinis et bien architectés et des cas d'utilisation qui effectuent des actions courantes à l'aide de services cloud de manière évolutive et sécurisée. Parce que Constructs fournit une bibliothèque pour les langages de programmation modernes, vous pouvez appliquer les compétences de développement existantes et les outils familiers à la tâche de construire une infrastructure cloud bien conçue pour vos solutions.

Les autres avantages d'AWS Solutions Constructs incluent :

- Il est basé sur le cloud AWS Development Kit (AWS CDK).
- Utilisez la logique (instructions if, for-loops, etc.) lors de la définition de l'infrastructure de votre solution.
- Utilisez des techniques orientées objet pour créer un modèle de votre système.
- Définissez des abstractions de haut niveau, partagez-les et publiez-les dans votre équipe, votre entreprise ou votre communauté.
- Organisez vos solutions en modules logiques.

- Partagez et réutilisez votre solution en tant que bibliothèque.
- Testez votre code d'infrastructure à l'aide de protocoles standard.
- Utilisez votre flux de travail de révision de code existant.

L'objectif d'AWS Solutions Constructs est de réduire la complexité et la logique de collage requise lors de l'intégration de modèles communs bien architectés afin d'atteindre vos objectifs de solution sur AWS.

Premiers pas avec les constructions AWS Solutions

Cette rubrique décrit comment installer et configurer AWS Cloud Development Kit (AWS CDK), AWS Solutions Constructions et créer votre première application AWS CDK à l'aide de modèles AWS Solutions Constructs.

> **Note**
>
> AWS Solutions Constructs est pris en charge sur les versions CDK AWS ≥ 1.46.0.

> **Tip**
>
> Tu veux creuser plus profondément ? Essayez leAtelier CDKpour une visite plus approfondie d'un projet réel.

> **Tip**
>
> Pour plus d'informations sur la mise en route avec le cloud AWS Development Kit (AWS CDK), consultez leManuel du développeur CDK AWS.

Prerequisites

AWS Solutions Constructs est basé sur le CDK AWS, vous devez donc installer Node.js (>= 10.3.0), même ceux qui travaillent dans des langues autres que TypeScript ou JavaScript. C'est parce que leCDK AWSet AWS Solutions Constructs sont développés dans TypeScript et exécutés sur Node.js. Les liaisons pour les autres langues prises en charge utilisent ce backend et ce jeu d'outils.

Vous devez fournir vos informations d'identification et une région AWS pour utiliser l'interface de ligne de commande AWS CDK, comme décrit dans Spécifier vos informations d'identification et votre région.

Les autres conditions préalables dépendent de votre langage de développement, comme suit.

Langage	Prérequis
P	Python >= 3.6
T t	Script TypeScript >= 2.7
Ja	Java >= 1.8

Installation du CDK AWS

Pour installer et configurer le CDK AWS, reportez-vous au Guide du développeur AWS CDK -Installation du CDK AWS.

Utilisation des constructions AWS Solutions

Le flux de travail typique pour la création d'une nouvelle application lorsque vous travaillez avec AWS Solutions Constructs suit la même approche que le CDK AWS.

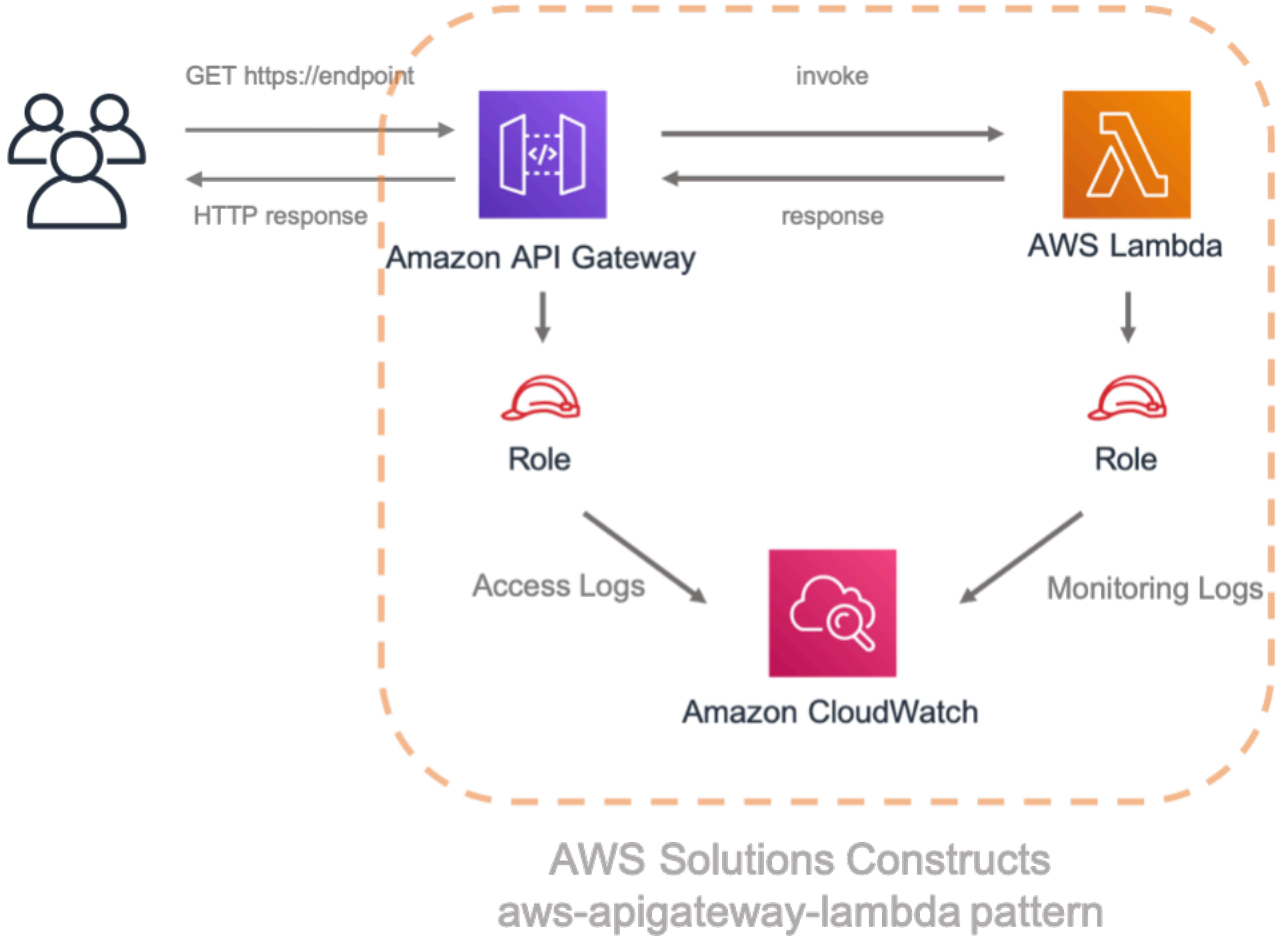
1. Créez le répertoire de l'application.
2. Initialiser.
3. Ajoutez les dépendances de modèle AWS Solutions Constructs.
4. Ajoutez du code supplémentaire à l'application.
5. Compilez l'application, si nécessaire.
6. Déployez les ressources définies dans l'application.
7. Tester l'application

S'il y a des problèmes, faites une boucle à travers modifier, compiler (si nécessaire), déployer et tester à nouveau.

Procédure pas à pas - Partie 1

> **Note**
>
> Constructions AWS Solutions est pris en charge sur les versions AWS CDK ≥ 1.46.0.

Ce didacticiel vous explique comment créer et déployer une application AWS CDK simple « Hello Constructs » qui utilise un modèle issu de AWS Solutions Constructs, de l'initialisation du projet au déploiement du modèle AWS CloudFormation résultant. L'application Hello Constructs va créer la solution simple suivante :



Hello Constructions

Commençons à créer notre première application AWS CDK à l'aide d'un développement basé sur des modèles.

> **Note**
>
> Ceci est un exemple de modification de`Hello CDK!`à partir desAtelier CDK. Si c'est la première fois que vous utilisez le CDK AWS, nous vous recommandons de commencer par cet atelier pour une procédure pas à pas pratique et comment tirer parti du CDK pour construire un projet réel.

Création de l'annuaire des applications et initialisation du CDK AWS

Créez un répertoire pour votre application CDK, puis créez une application AWS CDK dans ce répertoire.

TypeScript

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

Python

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

> **Tip**
>
> C'est le bon moment pour ouvrir le projet dans votre IDE préféré et explorer. Pour en savoir plus sur la structure du projet, sélectionnez le lien approprié :
>
> - TypeScript
> - Python

Modification des dépendances de base de projet

> **Warning**
>
> Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK doivent utiliser le même numéro de version dans votre projet. Par exemple, si vous utilisez AWS Solutions Constructs v.1.52.0, vous devez également utiliser AWS CDK v.1.52.0.

> **Tip**
>
> Prenez note de la version la plus récente d'AWS Solutions Constructs et appliquez ce numéro de version au`VERSION_NUMBER`dans les étapes ci-dessous (pour les conceptions AWS Solutions et les packages CDK AWS). Pour vérifier toutes les versions publiques de la bibliothèque Constructs,Cliquez ici.

TypeScript

Modifiez l'outil`package.json`avec les informations suivantes :

```
"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

Modifiez l'outil`setup.py`avec les informations suivantes :

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

Installez les dépendances de base des projets.

TypeScript

```
npm install
```

Python

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

Créez et exécutez l'application et confirmez qu'elle crée une pile vide.

TypeScript

```
npm run build
cdk synth
```

Python

```
cdk synth
```

Vous devriez voir une pile comme suit, où`CDK-VERSIO`Nest la version du CDK. (Votre sortie peut différer légèrement de ce qui est montré ici.)

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
```

Code de gestionnaire Lambda

Nous commencerons par le code du gestionnaire AWS Lambda.

Créer un annuaire`lambda`à la racine de votre arbre de projet.

TypeScript

Ajouter un fichier, appelé`lambda/hello.js`avec les éléments suivants :

```
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

Ajouter un fichier, appelé`lambda/hello.py`avec les éléments suivants :

```
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'])
    }
```

Ceci est une simple fonction Lambda qui renvoie le texte « Bonjour, Constructs ! Vous avez frappé [url path] ». La sortie de la fonction inclut également le code d'état HTTP et les en-têtes HTTP. Ceux-ci sont utilisés par API Gateway pour formuler la réponse HTTP à l'utilisateur.

Ce Lambda est fourni en JavaScript. Pour plus d'informations sur l'écriture des fonctions Lambda dans la langue de votre choix, reportez-vous à laDocumentation AWS Lambda.

Installer les dépendances AWS CDK et AWS Solutions Construit

Les constructions AWS Solutions sont livrées avec une vaste bibliothèque de constructions. La bibliothèque est divisée en modules, un pour chaque modèle bien conçu. Par exemple, si vous souhaitez définir une API Amazon API Gateway Rest vers une fonction AWS Lambda, nous devrons utiliser la méthode`aws-apigateway-lambda`Bibliothèque de modèles.

Nous devons également ajouter la bibliothèque de construction AWS Lambda et Amazon API Gateway à partir du CDK AWS.

Installez le module AWS Lambda et toutes ses dépendances dans notre projet :

> **Note**
>
> N'oubliez pas de remplacer la version correcte et correspondante à utiliser à la fois pour les constructions AWS Solutions et pour le CDK AWS dans le répertoire`VERSION_NUMBER`pour chaque commande. L'inadéquation des versions entre les packages peut entraîner des erreurs.

TypeScript

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

Python

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

Ensuite, installez le module Amazon API Gateway et toutes ses dépendances dans notre projet :

TypeScript

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

Python

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

Enfin, installez les constructions AWS Solutions`aws-apigateway-lambda`et toutes ses dépendances dans notre projet :

TypeScript

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

Python

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

Ajouter un modèle Amazon API Gateway/AWS Lambda à votre pile

Désormais, définissons le modèle AWS Solutions Constructs pour implémenter un Amazon API Gateway avec un proxy AWS Lambda.

TypeScript

Modification du fichier`lib/hello-constructs.ts`avec les éléments suivants :

```
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

Modification du fichier`hello_constructs/hello_constructs_stack.`pyavec les éléments suivants :

```
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
                )
            )
        )
```

C'est ça. C'est tout ce que vous devez faire pour définir une API Gateway qui envoie toutes les requêtes à une fonction AWS Lambda. Comparons notre nouvelle pile à celle d'origine :

TypeScript

```
npm run build
cdk diff
```

Python

```
cdk diff
```

La sortie doit se présenter comme suit :

```
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"},"/"]]}}
```

C'est sympa. Cet exemple simple avec un modèle bien conçu issu des constructions AWS Solutions a ajouté 21 nouvelles ressources à votre pile.

Déploiement sur cdk

> **Tip**
>
> Avant de pouvoir déployer votre première application AWS CDK contenant une fonction Lambda, vous devez amorcer votre environnement AWS. Cela crée un compartiment intermédiaire que le CDK AWS utilise pour déployer des piles contenant des ressources. Si c'est la première fois que vous utilisez le CDK AWS pour déployer des ressources, vous devrez exécuter le`cdk bootstrap`pour déployer la pile CDK Toolkit dans votre environnement AWS.

Ok, prêt à effectuer le déploiement ?

```
cdk deploy
```

Sortie de pile

Lorsque le déploiement est terminé, vous remarquerez cette ligne :

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

Il s'agit d'une sortie de pile qui est automatiquement ajoutée par le modèle AWS Solutions Constructs et inclut l'URL du point de terminaison API Gateway.

Tester votre application

Essayons de frapper ce point de terminaison avec`curl`. Copiez l'URL et exécutez (votre préfixe et votre région seront probablement différents).

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

Sortie doit se présenter comme suit :

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

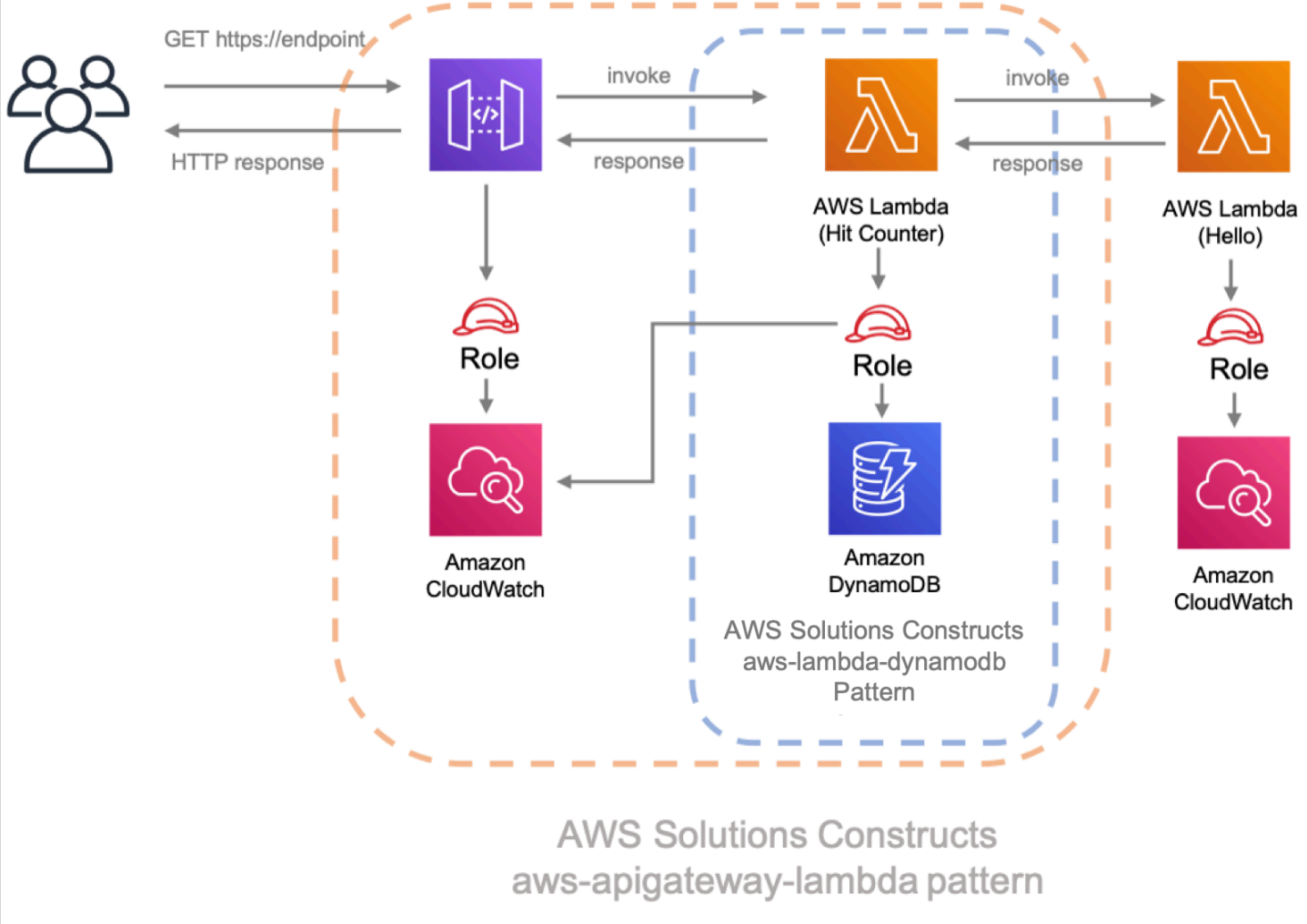
Si c'est la sortie que vous avez reçue, votre application fonctionne !

Procédure pas à pas - Partie 2

> **Note**
>
> AWS Solutions Constructions est pris en charge sur les versions CDK d'AWS ≥ 1.46.0.

Ce tutoriel vous explique comment modifier l'application « Hello Constructs » créée dansPART 1. Notre modification ajoutera un compteur de succès de site en utilisant AWS Lambda au modèle

DynamoDB d'AWS Solutions Constructs. La modification de l'application Hello Constructs entraînera la solution suivante :



Code de compteur Lambda

Commençons par écrire le code pour la fonction Hit Counter AWS Lambda. Cette fonction :

- incrémenter un compteur lié au chemin d'API dans une table Amazon DynamoDB,
- invoquez la fonction Hello AWS Lambda en aval,
- et renvoyer la réponse à l'utilisateur final.

TypeScript

Ajouter un fichier, appelé`lambda/hitcounter.js`avec les éléments suivants :

```
  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

Ajouter un fichier, appelé`lambda/hitcounter.py`avec les éléments suivants :

```
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)
```

Installer les nouvelles dépendances

> **Note**
>
> N'oubliez pas de remplacer la version correcte et correspondante à utiliser à la fois pour AWS Solutions Constructs et pour le CDK AWS dans le`VERSION_NUMBER`pour chaque commande. Ce numéro doit être identique au numéro de version utilisé pour les dépendances dans la première partie de cette procédure pas à pas. L'inadéquation des versions entre les packages peut entraîner des erreurs.

Comme d'habitude, nous devons d'abord installer les dépendances dont nous avons besoin pour la mise à jour de notre solution. Tout d'abord, il nous faut installer la bibliothèque de construction DynamoDB :

TypeScript

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

Python

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

Enfin, installez les constructions AWS Solutions`aws-lambda-dynamodb`et toutes ses dépendances dans notre projet :

TypeScript

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

Python

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

Définissez les ressources

Maintenant, mettons à jour notre code de pile pour accommoder notre nouvelle architecture.

Tout d'abord, nous allons importer nos nouvelles dépendances et déplacer la fonction « Hello » en dehors du`aws-apigateway-lambda`modèle que nous avons créé dans la partie 1.

TypeScript

Modifier le fichier`lib/hello-constructs.ts`avec les éléments suivants :

```
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);
  }
}
```

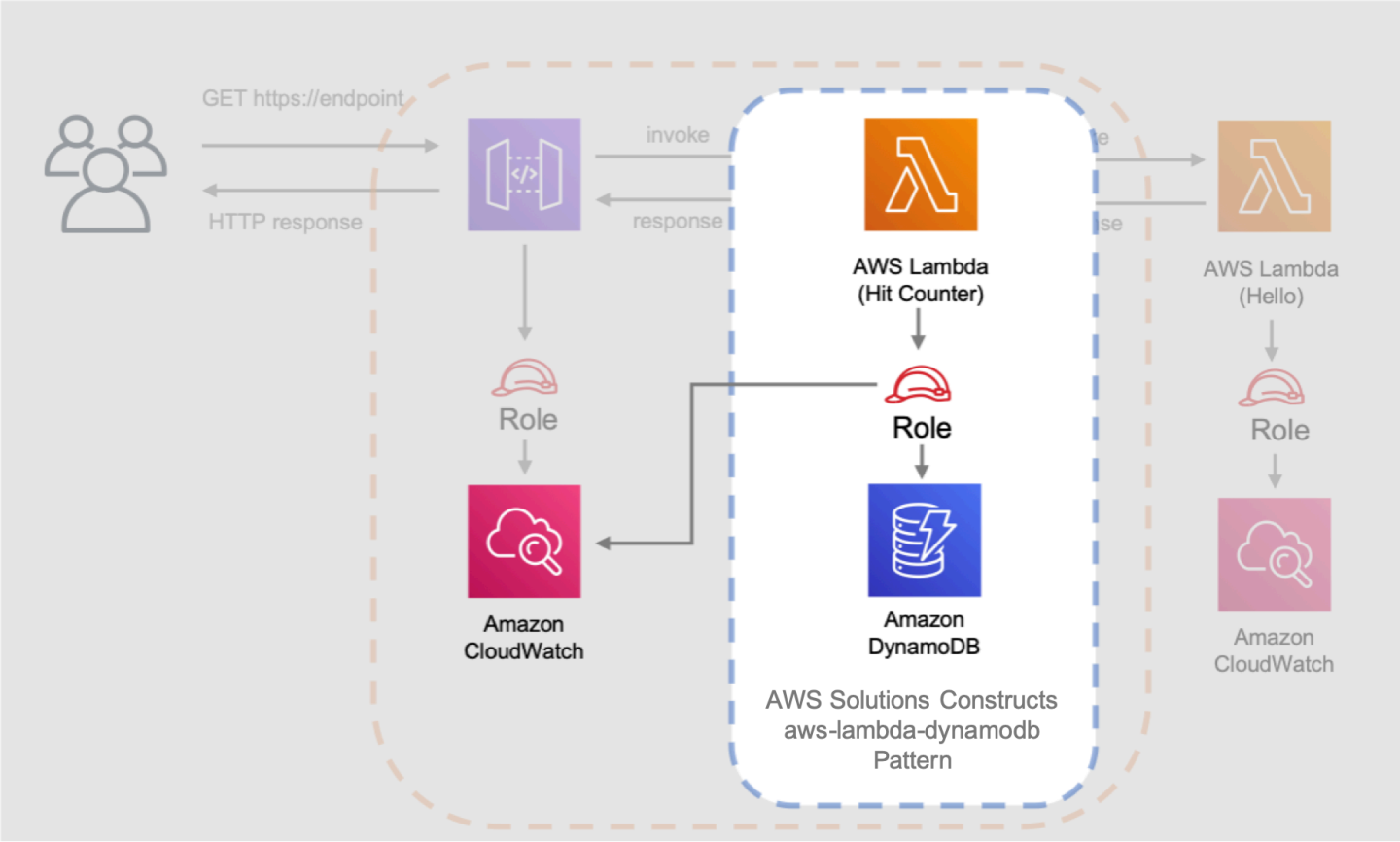
Python

Modifier le fichier`hello_constructs/hello_constructs_stack.py`avec les éléments suivants :

```
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
                )
            )
        )
```

Ensuite, nous allons ajouter le`aws-lambda-dynamodb`pour construire le service de comptoir à succès pour notre architecture mise à jour.



AWS Solutions Constructs
aws-apigateway-lambda pattern

La prochaine mise à jour ci-dessous définit les propriétés de la`aws-lambda-dynamodb`en définissant la fonction AWS Lambda avec le gestionnaire Hit Counter. En outre, la table Amazon DynamoDB est définie avec le nom`Hits`et une clé de partition de`path`.

TypeScript

Modifier le fichier`lib/hello-constructs.ts`avec les éléments suivants :

```
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

Modifier le fichier`hello_constructs/hello_constructs_stack.`pyavec les éléments suivants :

```
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
                )
            )
        )
```

Ensuite, il nous faut accorder la fonction Hit Counter créée à partir du paramètre`aws-lambda-dynamodb`ajouté ci-dessus la permission d'invoquer notre fonction Hello.

TypeScript

Modifier le fichier`lib/hello-constructs.ts`avec les éléments suivants :

```
  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

Modifier le fichier`hello_constructs/hello_constructs_stack.py`avec les éléments suivants :

```
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
                )
            )
        )
```

Enfin, nous devons mettre à jour notre`aws-apigateway-lambda`pour utiliser notre nouvelle fonction Hit Counter qui a été provisionnée avec le`aws-lambda-dynamodb`Modèle ci-dessus.

TypeScript

Modifier le fichier`lib/hello-constructs.ts`avec les éléments suivants :

```
  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

Modifier le fichier`hello_constructs/hello_constructs_stack.`pyavec les éléments suivants :

```
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
                )
            )
        )
```

Vérifiez les modifications

Construisons notre projet et examinons les changements apportés à nos ressources qui se produiront lorsque nous déployons ceci :

```
npm run build
cdk diff
```

Notre sortie doit se présenter comme suit :

```
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"
        [ ] ]
```

Déploiement CDK

OK, prêt à déployer ?

```
cdk deploy
```

Sortie de pile

Lorsque le déploiement est terminé, vous remarquerez cette ligne :

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

Tester votre application

Essayons de frapper ce point de terminaison avec curl. Copiez l'URL et exécutez (votre préfixe et votre région seront probablement différents).

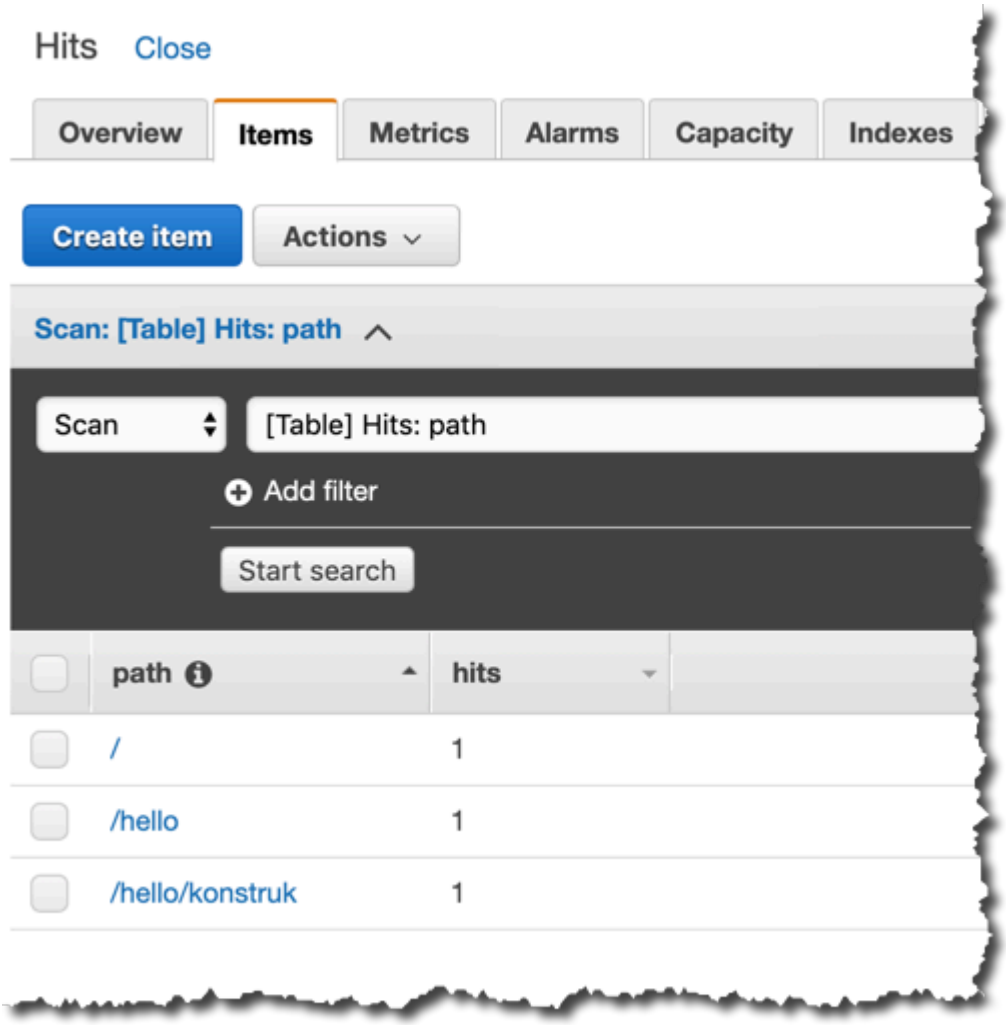
```
curl https://xxxxxxxxxx.execute-api.us-east-1.amazonaws.com/prod/
```

Sortie doit se présenter comme suit :

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

Maintenant, nous allons passer en revue les élémentsHitsTable Amazon DynamoDB.

1. Accédez à la console DynamoDB.
2. Vérifiez que vous êtes dans la région dans laquelle vous avez créé la table.
3. Tâche de sélectionTablesdans le panneau de navigation et sélectionnez l'optionHitsTable.
4. Ouvrez le tableau et sélectionnez « Articles ».
5. Vous devriez voir combien de coups vous avez obtenu pour chaque chemin.



6. Essayez d'accéder à un nouveau chemin et actualisez la vue Eléments. Vous devez voir un nouvel élément avec un`hits`comte d'un.

Si c'est la sortie que vous avez reçue, votre application fonctionne !

Exemple de cas d'utilisation

Cette bibliothèque comprend une collection d'implémentations de cas d'utilisation fonctionnelle pour démontrer l'utilisation des modèles architecturaux de Constructs. Ceux-ci peuvent être utilisés de la même manière que les modèles architecturaux, et peuvent être conceptualisés comme une abstraction supplémentaire de « plus haut niveau » de ces motifs. Les cas d'utilisation suivants sont fournis à titre d'exemples fonctionnels :

Site Web AWS Static S3

Ce modèle de cas d'utilisation (`aws-s3-static-website`) implémente une distribution Amazon CloudFront, un compartiment Amazon S3 et une ressource personnalisée basée sur AWS Lambda pour copier le contenu du site Web statique pour le site Web de démonstration Wild Rydes (faisant partie du`aws-serverless-web-app`Implémentation).

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

Gestionnaire d'images simple sans serveur AWS

Ce modèle de cas d'utilisation (`aws-serverless-image-handler`) implémente une distribution Amazon CloudFront, une API REST Amazon API Gateway, une fonction AWS Lambda et les autorisations/logiques nécessaires pour provisionner une API de gestionnaire d'images fonctionnelle pour diffuser du contenu d'image à partir d'un ou de plusieurs compartiments Amazon S3 dans le compte de déploiement.

> Code source (gestionnaire d'images aws-serverless-image-handler)
>
> https://github.com/awslabs/aws-solutions-constructs/tree/master/source/use_cases/aws-serverless-image-handler

AWS plémentation web sans serveur

Ce modèle de cas d'utilisation (`aws-serverless-web-app`) implémente une application web simple sans serveur qui permet aux utilisateurs de demander des promenades en licorne à la flotte Wild Rydes. L'application présentera aux utilisateurs une interface utilisateur basée sur HTML pour indiquer l'emplacement où ils aimeraient être récupérés et interfacera sur le backend avec un service Web RESTful pour soumettre la demande et envoyer une licorne à proximité. L'application fournira également des facilités aux utilisateurs pour s'inscrire au service et se connecter avant de demander des promenades.

Code source (aws-serverless-web-app)

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

Référence API

AWS Solutions Constructs (Constructs) est une extension open source d'AWS Cloud Development Kit (AWS CDK) qui fournit des modèles multi-services et bien architectés pour définir rapidement des solutions dans le code afin de créer une infrastructure prévisible et reproductible. L'objectif de Constructs est d'accélérer l'expérience des développeurs pour construire des solutions de n'importe quelle taille en utilisant des définitions basées sur des motifs pour leur architecture.

Les modèles définis dans Constructs sont des abstractions multi-services de haut niveau des constructions AWS CDK qui ont des configurations par défaut basées sur des meilleures pratiques bien conçues. La bibliothèque est organisée en modules logiques utilisant des techniques orientées objet pour créer chaque modèle de modèle architectural.

Le kit CDK est disponible dans les langues suivantes :

- JavaScript, TypeScript (Node.js ≥ 10.3.0)
- Python (Python ≥ 3,6)
- Java (Java ≥ 1,8)

Modules

AWS Solutions Constructs est organisé en plusieurs modules. Ils sont nommés comme ceci :

- aws-xxx : Paquet de motifs bien conçu pour les services indiqués. Ce package contiendra des constructions qui contiennent plusieurs modules de service AWS CDK pour configurer le modèle donné.
- xxx : Paquets qui ne démarrent pas »aws-« sont des modules de base de Constructs qui sont utilisés pour configurer les meilleures pratiques par défaut pour les services utilisés dans la bibliothèque de modèles.

Contenu du module

Les modules contiennent les types suivants :

- Modèles- Toutes les constructions multi-services de niveau supérieur dans cette bibliothèque.
- Autres types- Toutes les classes non construites, interfaces, structures et énumérations qui existent pour prendre en charge les modèles.

Les modèles prennent un ensemble de propriétés (en entrée) dans leur constructeur ; l'ensemble des propriétés (et celles qui sont requises) peut être vu sur la page de documentation d'un modèle.

La page de documentation du modèle répertorie également les méthodes disponibles à appeler et les propriétés qui peuvent être utilisées pour récupérer des informations sur le modèle après son instanciation.

aws-apigateway-dynamodb

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion sémantique de versionModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_apigateway_dynamodb`
Typecript	`@aws-solutions-constructs/aws-apigateway-dynamodb`
Java	`software.amazon.awsconstructs.services.apigatewaydynamodb`

Overview

Cette AWS Solutions Construct implémente une API REST Amazon API Gateway connectée à une table Amazon DynamoDB.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`ApiGatewayToDynamoDBProps`

Modèle Construire des accessoires

Nom	Type	Description
DynamoTableProps	`dynamodb.TableProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour DynamoDB Table
ApigatewayProps ?	`api.RestApiProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de la API Gateway.
AllowCreateOperation	`boolean`	Indique s'il faut déployer API Gateway Method for

Nom	Type	Description
		Create opération sur la table DynamoDB.
CreateRequestTemplate	`string`	Modèle de demande de API Gateway pour la méthode Create, obligatoire si AllowCreateOperation défini sur true
AllowReadOpération	`boolean`	Indique s'il faut déployer API Gateway Method for Read opération sur la table DynamoDB.
AllowUpdateOpération	`boolean`	Que ce soit pour déployer la méthode API Gateway pour l'opération de mise à jour sur la table DynamoDB.
UpdateRequestTemplate	`string`	Modèle de demande de API Gateway pour la méthode Update, requis si allowUpdateOperation est défini sur true
AllowDeleteOpération	`boolean`	Indique s'il faut déployer API Gateway Method for Delete opération sur la table DynamoDB.
LogGroupProps ?	`logs.LogGroupProps`	Des accessoires fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour le groupe de journaux CloudWatch Logs.

Propriétés du modèle

Nom	Type	Description
Apigateway	`api.RestApi`	Renvoie une instance de l'API Gateway API créée par le modèle.
ApigateWayCloudWatchRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle qui active la journalisation des accès à partir de l'API Gateway API vers CloudWatch.
ApigateWayLogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle auquel les journaux d'accès API REST de API Gateway d'API sont envoyés.
ApigateWayRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle pour l'API Gateway API.
DynamoTable	`dynamodb.Table`	Renvoie une instance de la table DynamoDB créée par le modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Amazon API Gateway

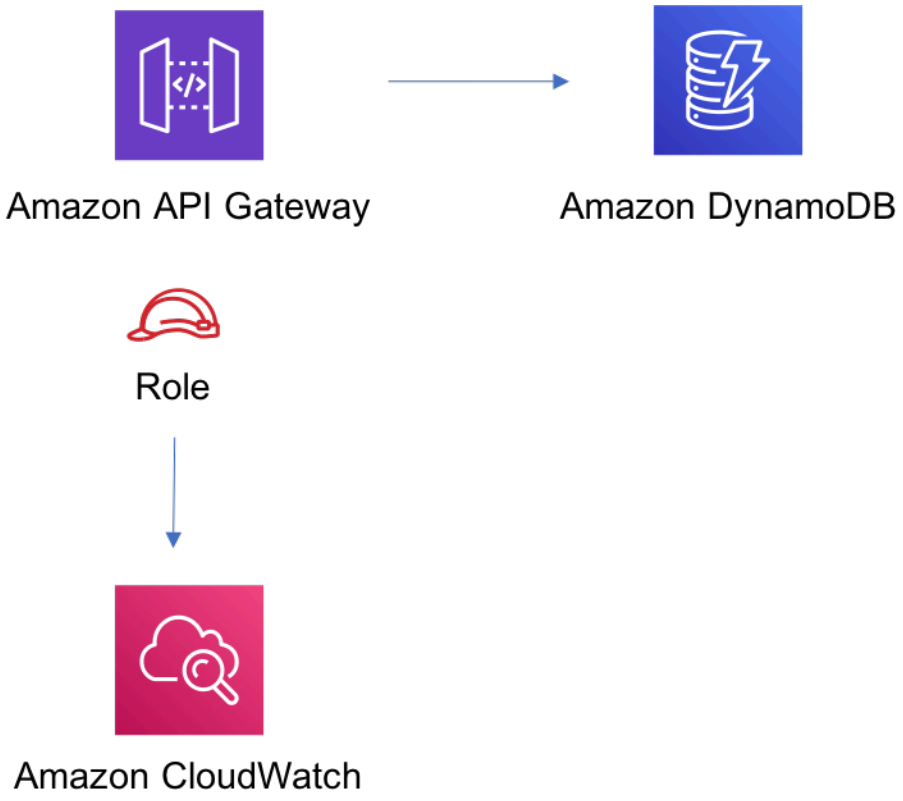
- Déployer un point de terminaison API optimisé pour les périphériques
- Activer la journalisation CloudWatch pour API Gateway

- Configurer le rôle IAM d'accès minimal aux privilèges pour API Gateway
- Définissez l'AuthorizationType par défaut pour toutes les méthodes d'API sur IAM
- Activer le suivi X-Ray

Amazon DynamoDB Table

- Définir le mode de facturation de la table DynamoDB à la demande (Paiement par demande)
- Activer le chiffrement côté serveur pour la table DynamoDB à l'aide de la clé KMS gérée par AWS
- Crée une clé de partition appelée 'id' pour la table DynamoDB
- Conserver la table lors de la suppression de la pile CloudFormation
- Activer les sauvegardes continues et la restauration à un instant dans le passé

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-constructs/aws-apigateway-dynamodb

aws-apigateway-iot

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Ceux-ci ne sont pas assujettis à laGestion sémantiqueModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Ce modèle AWS Solutions Construct implémente une API REST Amazon API Gateway connectée au modèle AWS IoT.

Cette construction crée un proxy HTTPS évolutif entre API Gateway et AWS IoT. Cela est utile lorsque vous souhaitez autoriser les périphériques hérités qui ne prennent pas en charge le protocole MQTT ou MQTT/WebSocket à interagir avec la plate-forme AWS IoT.

Cette implémentation permet de publier des messages en écriture seule sur des rubriques MQTT données, et prend également en charge les mises à jour instantanées des périphériques HTTPS pour autoriser des éléments dans le registre des périphériques. Il n'implique pas les fonctions Lambda pour la transmission de messages par proxy, et s'appuie plutôt sur l'intégration directe API Gateway vers AWS IoT qui prend en charge les messages JSON ainsi que les messages binaires.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`ApiGatewayToIotProps`

Modèle de construction des accessoires

Nom	Type	Description
IoTendpoint	`string`	Le sous-domaine de point de terminaison AWS IoT pour intégrer la API Gateway avec (par exemple a12345678 90123-ats).
ApigateWayCreateApiKey ?	`boolean`	Si défini sur`true`, une clé API est créée et associée à un UsagePlan. L'utilisateur doit spécifier l'en-tête `x-api-ke y` lors de l'accès à RestApi. Valeur par défaut définie sur`false`.
ApigatewayExecutionRole ?	`iam.Role`	Rôle IAM utilisé par API Gateway pour accéder à AWS IoT. Si ce n'est pas spécifié, un rôle par défaut est créé avec un accès générique ('*') à tous les sujets et objets.
ApigatewayProps ?	`api.restApiProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de l'API Gateway API.
LogGroupProps ?	`logs.LogGroupProps`	Options fournies par l'utilisa teur pour remplacer les accessoires par défaut pour le groupe de journaux CloudWatch Logs.

Propriétés du modèle

Nom	Type	Description
AppiGateway	`api.RestApi`	Renvoie une instance de l'API REST API Gateway créée par le modèle.
ApigateWayCloudWatchRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle qui active la journalisation des accès à partir de l'API Gateway API vers CloudWatch.
ApigateWayLogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle auquel les journaux d'accès API REST de API Gateway d'API sont envoyés.
ApigateWayRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle pour l'API Gateway API.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Amazon API Gateway

- Déployer un point de terminaison API optimisé pour les périphériques
- Crée des ressources API avec`POST`Méthode de publication des messages dans des rubriques IoT
- Crée des ressources API avec`POST`Méthode de publication des messages dans`ThingShadow`and`NamedShadows`
- Activez la journalisation CloudWatch pour API Gateway

- Configurer le rôle IAM pour API Gateway avec accès à toutes les rubriques et objets
- Définissez l'AuthorizationType par défaut pour toutes les méthodes d'API sur IAM
- Activez le suivi X-Ray Tray
- Crée un UsagePlan et s'associe à`prod`stage

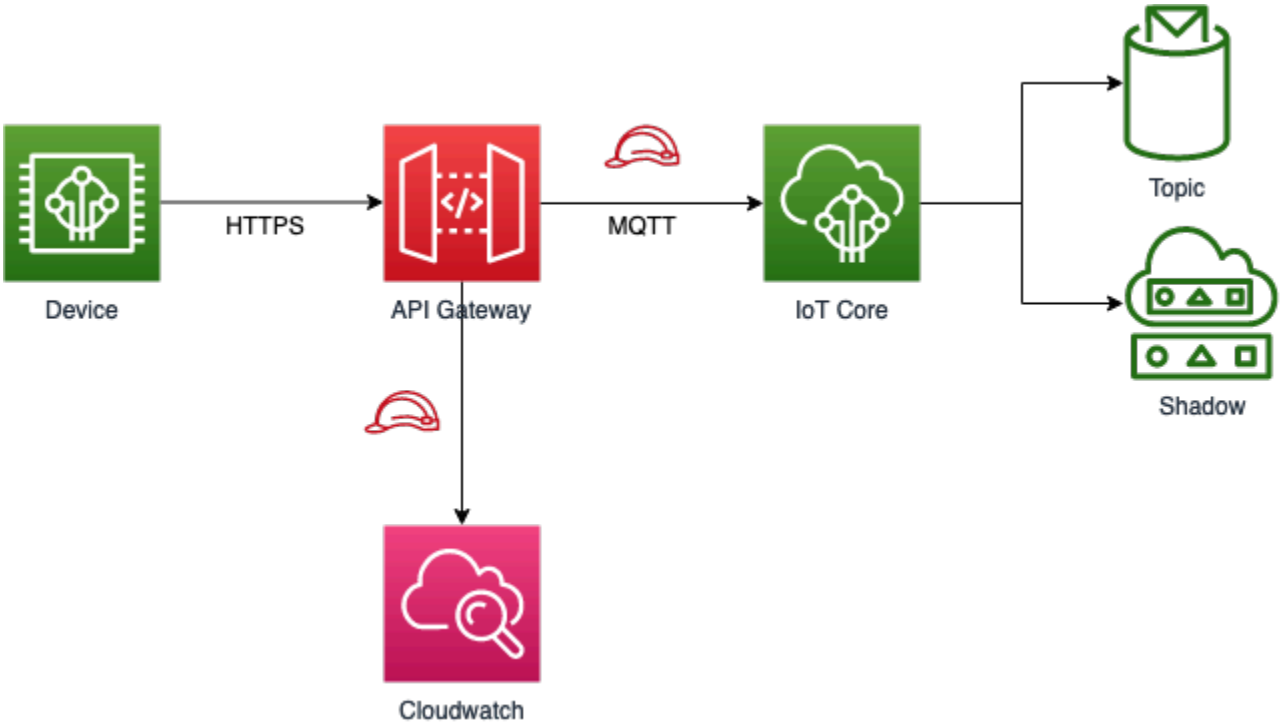
Vous trouverez ci-dessous une description des différentes ressources et méthodes exposées par API Gateway après le déploiement de Construct. ConsultezExemplespour plus d'informations sur la façon de tester facilement ces terminaux à l'aide de`curl`.

Méthode	Ressource	Paramètre (s) de requête	Code (s) renvoyé (s)	Description
`POST`	`/message/<topics>`	`qos`	`200/403/500`	En appelant ce point de terminaison, vous devez transmettre les rubriques sur lesquelles vous souhaitez publier (par exemple, `` `/message/device/foo` ``).
`POST`	`/shadow/<thingName>`	Aucune	`200/403/500`	Cette route permet de mettre à jour le document fictif d'une chose, compte tenu de son`thingName` à l'aide de type shadow non nommé (classique). La caisse doit être conforme

Méthode	Ressource	Paramètre (s) de requête	Code (s) renvoyé (s)	Description
				à la structure d'ombre standard comprenant un`statenoeud` et associé`desiredandrepo` ConsultezMise à jour des ombres de périphéri quepour obtenir un exemple.

Méthode	Ressource	Paramètre (s) de requête	Code (s) renvoyé (s)	Description
POST	/shadow/<thingName>/<shadowName>	Aucune	200/403/500	Cette route permet de mettre à jour le document d'ombre nommé d'une chose, compte tenu de sonthingName et l'shadowName à l'aide du type Ombre nommé. La caisse doit être conforme à la structure d'ombre standard comprenant unstatenoeud et associédesiredandrepo ConsultezMise à jour des ombres nomméespour obtenir un exemple.

Architecture



Examples

Les exemples suivants fonctionnent uniquement avec`API_KEY`, puisque l'autorisation IAM nécessite également un jeton SigV4 pour être spécifié, assurez-vous que le`apiGatewayCreateApiKey`de vos accessoires Construct est définie sur`true`lors du déploiement de la pile, sinon les exemples ci-dessous ne fonctionneront pas.

Publication d'un message

Vous pouvez utiliser`curl`pour publier un message sur différentes rubriques MQTT à l'aide de l'API HTTPS. L'exemple ci-dessous affichera un message sur le`device/foo`Rubrique.

```
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"}'
```

Remarque: Remplacez par.`stage-id`,`region`, et`api-key`avec vos valeurs de déploiement.

Vous pouvez enchaîner les noms de rubriques dans l'URL et l'API accepte jusqu'à 7 sous-rubriques sur lesquelles vous pouvez publier. Par exemple, l'exemple ci-dessous publie un message dans la rubrique`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"}'
```

Mise à jour des ombres de périphérique

Pour mettre à jour le document instantané associé à une chose donnée, vous pouvez émettre une demande d'état instantané en utilisant un nom de chose. consultez suivante sur la façon de mettre à jour un shadow de chose.

```
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" }}}'
```

Mise à jour des ombres nommées

Pour mettre à jour le document instantané associé à l'ombre nommée d'une chose donnée, vous pouvez émettre une demande d'état d'ombre à l'aide d'un nom de chose et d'un nom d'ombre. Reportez-vous à l'exemple suivant sur la mise à jour d'une ombre nommée.

```
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" }}}'
```

Envoi des charges utiles binaires

Il est possible d'envoyer une charge utile binaire à l'API proxy, jusqu'au service AWS IoT. Dans l'exemple suivant, nous envoyons le contenu de l'`README.md`associé à ce module (traité comme une donnée binaire) à`device/foo`à l'aide de l'`application/octet-stream`Type de contenu.

```
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
```

Remarque: Exécutez cette commande dans le répertoire de ce projet. Vous pouvez ensuite tester l'envoi d'un autre type de fichiers binaires à partir de votre système de fichiers.

GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-constructs/aws-apigateway-iot

aws-apigateway-kinesisstreams

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Ceux-ci ne sont pas assujettis à laGestion sémantique de versionsModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_apigateway_kinesisstreams`
TS	`@aws-solutions-constructs/aws-apigateway-kinesisstreams`

Langage	Package
Typecript	
Java	`software.amazon.awsconstructs.services.apigatewaykinesisstreams`

Overview

Ce modèle implémente une API REST Amazon API Gateway connectée à un flux de données Amazon Kinesis.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`ApiGatewayToKinesisStreamsProps`

Accessoires de construction de modèle

Nom	Type	Description
ApigatewayProps ?	`api.RestApiProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de l'API Gateway API.
PutRecordRequestTemplate ?	`string`	Modèle de demande de API Gateway pour l'action PutRecord. Si ce n'est pas fourni, une valeur par défaut sera utilisée.
PutRecordRequestModel ?	`api.ModelOptions`	Modèle de requête API Gateway pour l'action PutRecord. Si ce n'est pas fourni, une valeur par défaut sera créée.
PutRecordsRequestTemplate ?	`string`	Modèle de demande de API Gateway pour l'action PutRecords. Si ce n'est pas fourni, une valeur par défaut sera utilisée.
PutRecordRequestModel ?	`api.ModelOptions`	Modèle de requête API Gateway pour l'action PutRecords. Si ce n'est pas fourni, une valeur par défaut sera créée.
L'existence de Streamobj ?	`kinesis.Stream`	Instance existante de Kinesis Stream, fournissant à la fois ceci et`kinesisSt`

Nom	Type	Description
		`reamProps` provoquera une erreur.
KinesisStreamProps ?	`kinesis.StreamProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour le flux Kinesis.
LogGroupProps ?	`logs.LogGroupProps`	Exemples facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut pour le groupe de journaux CloudWatch Logs.

Propriétés de modèle

Nom	Type	Description
Apigateway	`api.RestApi`	Renvoie une instance de l'API Gateway API créée par le modèle.
ApigateWayRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle pour l'API Gateway API.
ApigateWayCloudWatchRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle qui active la journalisation des accès à partir de l'API Gateway API vers CloudWatc h.
ApigateWayLogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par

Nom	Type	Description
		le modèle auquel les journaux d'accès API REST de API Gateway d'API sont envoyés.
KinesisStream	`kinesis.Stream`	Renvoie une instance du flux Kinesis créé par le modèle.

Exemple d'utilisation de l'API

Méthode	Chemin de la demande	Corps de la demande	Action de la file	Description
POST	`/record`	`{ "data": "Hello World!", "partitionKey": "pk001" }`	`kinesis:PutRecord`	Écrit un seul enregistrement de données dans le flux.
POST	`/records`	`{ "records": [ { "data": "abc", "partitionKey": "pk001" }, { "data": "xyz",`	`kinesis:PutRecords`	Écrit plusieurs enregistrements de données dans le flux en un seul appel.

Méthode	Chemin de la demande	Corps de la demande	Action de la file	Description
		`"partitionKey": "pk001" } ] }`		

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

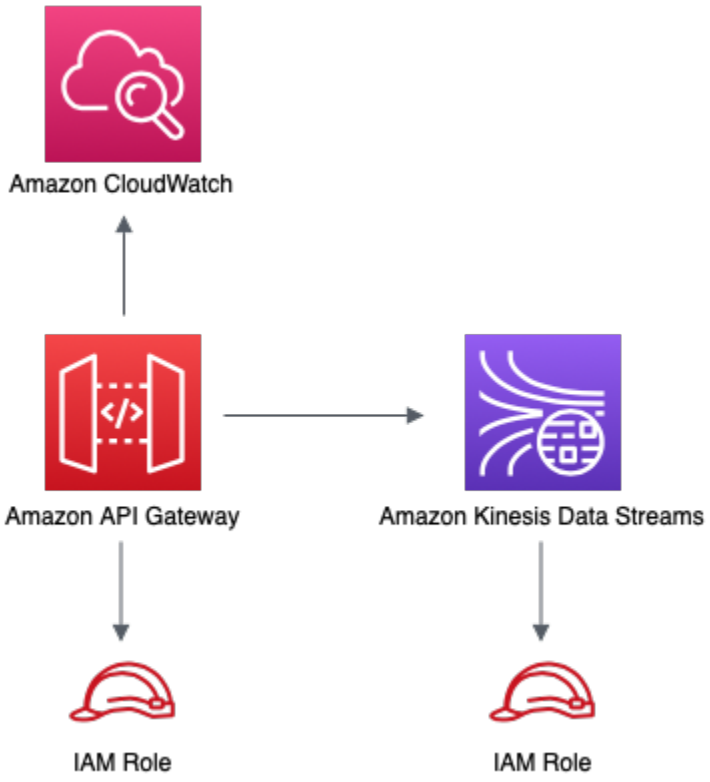
Amazon API Gateway

- Déployez un point de terminaison API optimisé pour les périphériques.
- Activez la journalisation CloudWatch pour API Gateway.
- Configurez le rôle IAM d'accès le moins privilégié pour API Gateway.
- Définissez l'AuthorizationType par défaut pour toutes les méthodes d'API sur IAM.
- Activer le suivi X-Ray.
- Valider le corps de la requête avant de transmettre des données à Kinesis.

Amazon Kinesis Data Stream

- Configurez le rôle IAM d'accès le moins privilégié pour le flux Kinesis.
- Activez le chiffrement côté serveur pour Kinesis Stream à l'aide de la clé KMS gérée AWS.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructs/aws-apigateway-kinesisstreams

aws-apigateway-lambda

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion sémantiqueModèle. Cela signifie que même si vous pouvez les utiliser, vous

devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

AWS Solutions Construct implémente une API REST Amazon API Gateway connectée à une fonction AWS Lambda.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`ApiGatewayToLambdaProps`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.
ApigatewayProps ?	`api.LambdaRestApiProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de l'API.
LogGroupProps ?	`logs.LogGroupProps`	Des accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires

Nom	Type	Description
		par défaut pour le groupe de journaux CloudWatch Logs.

Propriétés de modèle

Nom	Type	Description
ApigateWayCloudWatchRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle qui active la journalisation des accès à partir de l'API Gateway API vers CloudWatch.
ApigateWayLogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle auquel les journaux d'accès API REST de API Gateway d'API sont envoyés.
LambdaFunction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.
AppiGateway	`api.LambdaRestApi`	Renvoie une instance de l'API Gateway API créée par le modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

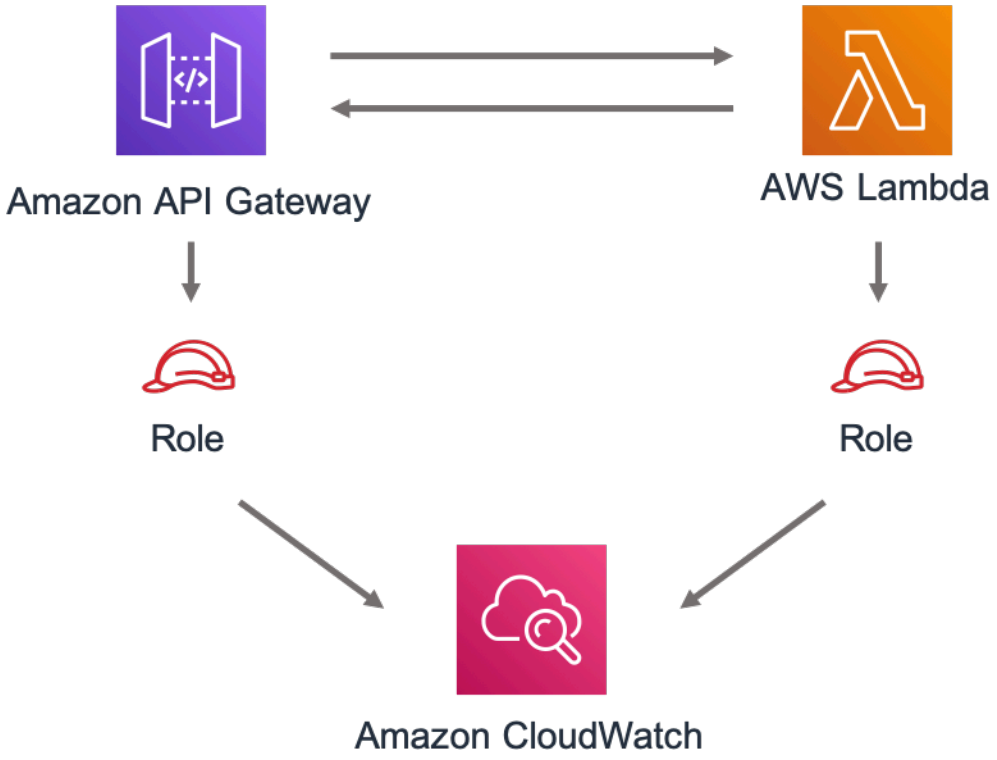
Amazon API Gateway

- Déploiement d'un terminal d'API optimisé pour les périphériques
- Activer la journalisation CloudWatch pour API Gateway
- Configurer le rôle IAM d'accès minimal aux privilèges pour API Gateway
- Définissez l'AuthorizationType par défaut pour toutes les méthodes d'API sur IAM
- Activer le suivi X-Ray
- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Fonction AWS Lambda

- Configuration du rôle IAM d'accès limité pour la fonction Lambda
- Activer la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda
- Activer le suivi X-Ray

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-construction/aws-apigateway-lambda

aws-apigateway-sagemakerendpoint

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion sémantique des versionsmodèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

AWS Solutions Construct implémente une API REST Amazon API Gateway connectée à un point de terminaison Amazon SageMaker.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`ApiGatewayToSageMakerEndpointProps`

Accessoires de construction de modèle

Nom	Type	Description
ApigatewayProps ?	`api.RestApiProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de l'API Gateway API.
ApigatewayExecutionRole ?	`iam.Role`	Rôle IAM utilisé par API Gateway pour appeler le point de terminaison SageMaker. Si ce n'est pas spécifié, un rôle par défaut est créé avec accès à`endpointName` .
EnpointName	`string`	Nom du point de terminaison d'inférence SageMaker déployé.
ResourceName ?	`string`	Nom de ressource facultatif où la méthode GET sera disponible.
chemin de la ressource	`string`	Chemin de la ressource pour la méthode GET. La variable définie ici peut être référencée dans`requestMappingTemplate` .
RequestMappingTemplate	`string`	Modèle de mappage pour convertir les requêtes GET reçues sur l'API REST en

Nom	Type	Description
		requêtes POST attendues par le point de terminaison SageMaker.
ResponseMappingTemplate ?	`string`	Modèle de mappage facultatif pour convertir les réponses reçues du point de terminaison SageMaker.
LogGroupProps ?	`logs.LogGroupProps`	Exemples facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut pour le groupe de journaux CloudWatch Logs.

Propriétés du modèle

Nom	Type	Description
Apigateway	`api.LambdaRestApi`	Renvoie une instance de l'API Gateway API créée par le modèle.
ApigateWayRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle pour l'API Gateway API.
ApigateWayCloudWatchRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle qui active la journalisation des accès à partir de l'API Gateway API vers CloudWatch.

Nom	Type	Description
ApigateWayLogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle auquel les journaux d'accès API REST de API Gateway d'API sont envoyés.

Exemple d'utilisation de l'API

Remarque: Chaque point de terminaison SageMaker est unique et la réponse de l'API dépend du modèle déployé. L'exemple donné ci-dessous suppose que l'échantillon decet article de blog. Pour obtenir une référence sur la façon dont cela serait mis en œuvre, veuillez vous référer àinteg.apigateway-sagemakerendpoint-overwrite.ts.

Méthode	Chemin de la demande	Chaîne de requête	Action SageMaker	Description
GET	`/321`	`items=101,131,162`	`sagemaker:InvokeEndpoint`	Récupère les prédictions d'un utilisateur et d'éléments spécifiques.

Paramètres par défaut

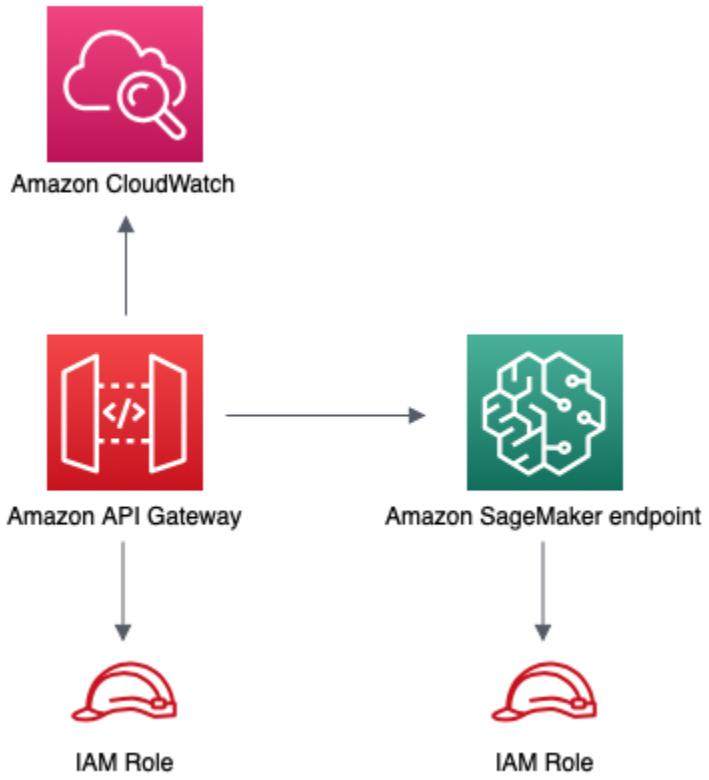
L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Amazon API Gateway

- Déployer un point de terminaison API optimisé pour les périphériques
- Activer la journalisation CloudWatch pour API Gateway
- Configurer le rôle IAM d'accès minimal aux privilèges pour API Gateway
- Définissez l'AuthorizationType par défaut pour toutes les méthodes d'API sur IAM
- Activer le suivi X-Ray

- Valider les paramètres de demande avant de transmettre les données à SageMaker

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructs/aws-apigateway-sagemakerendpoint

aws-apigateway-sqs

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion de versions sémantiquesmodèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

AWS Solutions Construct implémente une API REST Amazon API Gateway connectée à une file d'attente Amazon SQS.

Voici une définition de modèle déployable minimale dans TypeScript :

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

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

Initializer

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

Paramètres

- scopeConstruct
- idstring
- propsApiGatewayToSqsProps

Accessoires de construction de modèle

Nom	Type	Description
ApigatewayProps ?	api.RestApiProps	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de la API Gateway.
QueueProps ?	sqs.QueueProps	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de la file d'attente.
Déploiement Deadlette rQueue ?	boolean	Indique s'il faut déployer une file d'attente secondaire à utiliser comme file d'attente de lettres mortes. La valeur par défaut est true.
MaxReceiveCount	number	Nombre de fois qu'un message peut être défile d'attente sans succès avant d'être déplacé vers la file d'attente de lettres mortes.
AllowCreateOperation ?	boolean	Indique s'il faut déployer une méthode API Gateway pour les opérations Create sur la

Nom	Type	Description
		file d'attente (par exemple SQS:SendMessage).
CreateRequestTemplate ?	`string`	Remplacer le modèle de demande API Gateway par défaut pour la méthode Create, si `allowCreateOperation` a la valeur `true`.
AllowreadOperation ?	`boolean`	Indique s'il faut déployer une méthode API Gateway pour les opérations Read sur la file d'attente (par exemple SQS:ReceiveMessage).
ReadRequestTemplate ?	`string`	Remplacer le modèle de requête API Gateway par défaut pour la méthode Read, si `allowReadOperation` a la valeur `true`.
AllowDeleteOperation ?	`boolean`	Indique s'il faut déployer une méthode API Gateway pour les opérations Delete sur la file d'attente (c'est-à-dire SQS:DeleteMessage).
DeleteRequestTemplate ?	`string`	Remplacer le modèle de demande API Gateway par défaut pour la méthode Delete, si `allowDeleteOperation` a la valeur `true`.

Nom	Type	Description
LogGroupProps ?	`logs.LogGroupProps`	Accessoires fournis par l'utilisateur pour remplacer les accessoires par défaut du groupe de journaux CloudWatch Logs.

Propriétés du modèle

Nom	Type	Description
AppiGateway	`api.RestApi`	Renvoie une instance de l'API Gateway API créée par le modèle.
ApigateWayCloudWatchRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle qui active la journalisation des accès à partir de l'API Gateway API vers CloudWatch.
ApigateWayLogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle auquel les journaux d'accès API REST de API Gateway d'API sont envoyés.
ApigateWayRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle pour l'API Gateway API.
DeadletterQueue ?	`sqs.Queue`	Renvoie une instance de la file d'attente de lettres mortes

Nom	Type	Description
		créée par le modèle, si une instance est déployée.
SQSqueue	`sqs.Queue`	Renvoie une instance de la file d'attente SQS créée par le modèle.

Exemples de scénario d'utilisation

Méthode	Chemin de la demande	Corps de la demande	Action de la file	Description
GET	`/`		`sqs::ReceiveMessage`	Récupère un message de la file d'attente.
POST	`/`	`{ "data": "Hello World!" }`	`sqs::SendMessage`	Remet un message dans la file d'attente.
DELETE	`/message?receiptHandle=[value]`		`sqs::DeleteMessage`	Supprime un message spécifié de la file d'attente

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Amazon API Gateway

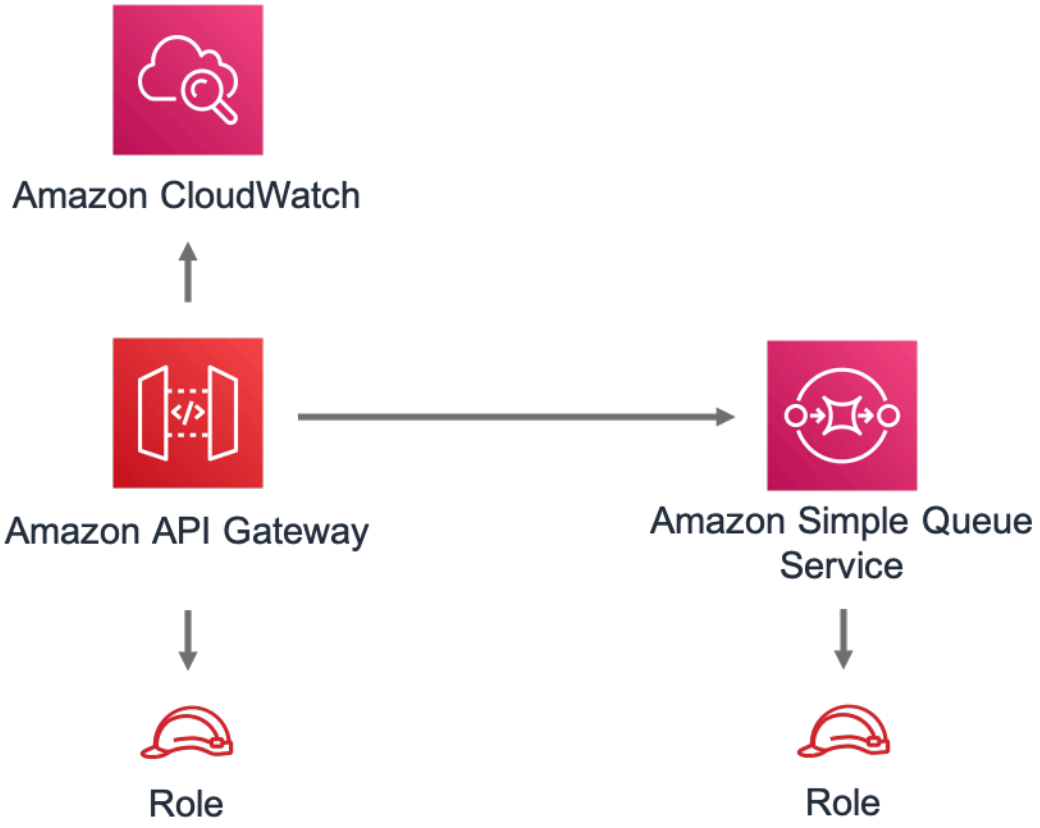
- Déployer un point de terminaison API optimisé pour les périphériques
- Activer la journalisation CloudWatch pour API Gateway
- Configurer le rôle IAM d'accès minimal aux privilèges pour API Gateway

- Définissez l'AuthorizationType par défaut pour toutes les méthodes d'API sur IAM
- Activer le suivi X-Ray

File d'attente Amazon SQS

- Déployer la file d'attente de lettres mortes SQS pour la file d'attente SQS source
- Activer le chiffrement côté serveur pour la file d'attente SQS source à l'aide de la clé KMS gérée par AWS
- Application du chiffrement des données en transit

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-constructs/aws-apigateway-sqs

aws-cloudfront-apigateway

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion sémantiqueModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Cette solution AWS Construct implémente une distribution Amazon CloudFront devant une API REST Amazon API Gateway.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scopeConstruct
- idstring
- propsCloudFrontToApiGatewayProps

Accessoires de construction

Nom	Type	Description
ExistingApigatewayObj	api.RestApi	La API Gateway régionale qui sera frontée avec CloudFront
CloudFrontDistributionProps ?	cloudfront.DistributionProps	Des accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut pour la distribution CloudFront.
InserThttpSecurityHeaders ?	boolean	Props fournis par l'utilisateur en option pour activer/désactiver l'injection automatique des en-têtes de sécurité HTTP des meilleures pratiques dans toutes les réponses de CloudFront

Propriétés de modèle

Nom	Type	Description
ApiGateway	`api.RestApi`	Renvoie une instance de l'API Gateway API créée par le modèle.
CloudFrontLoggingBucket ?	`s3.Bucket`	Renvoie une instance du compartiment de journalisation créé par le modèle pour la distribution Web CloudFront.
CloudFrontWebDistribution	`cloudfront.CloudFrontWebDistribution`	Renvoie une instance de la distribution Web CloudFront créée par le modèle.
EdgelAmbDAFunction Version ?	`lambda.Version`	Renvoie une instance de la version de la fonction de bord Lambda créée par le motif.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

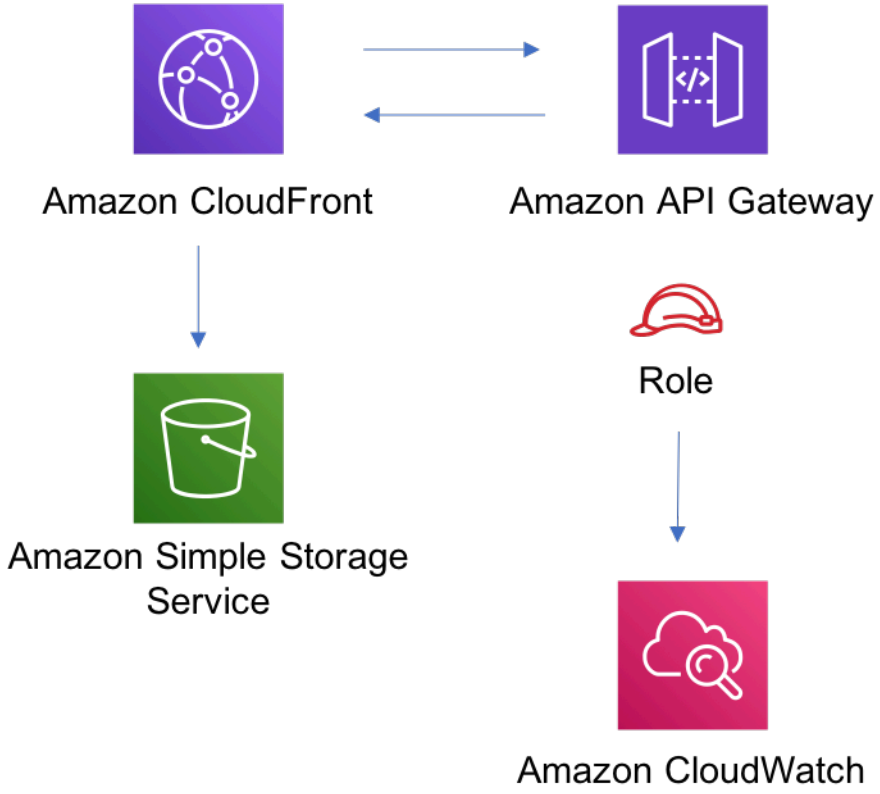
Amazon CloudFront

- Configurer la journalisation d'accès pour CloudFront WebDistribution
- Activer l'injection automatique des en-têtes de sécurité HTTP des meilleures pratiques dans toutes les réponses de CloudFront WebDistribution

Amazon API Gateway

- L'objet API Gateway fourni par l'utilisateur est utilisé tel quel
- Activer le suivi X-Ray

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructs/aws-cloudfront-apigateway

aws-cloudfront-apigateway-lambda

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion sémantique de versionModèle. Cela signifie que même si vous pouvez les utiliser, vous

devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Cette solution AWS Construct implémente une distribution Amazon CloudFront devant une API REST basée sur Amazon API Gateway Lambda.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`CloudFrontToApiGatewayToLambdaProps`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.
ApigatewayProps ?	`api.LambdaRestApiProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour API Gateway
CloudFrontDistributionProps ?	`cloudfront.DistributionProps`	Des accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires

Nom	Type	Description
		par défaut pour la distribution CloudFront.
InserThttpSecurityHeaders ?	`boolean`	Props fournis par l'utilisa teur en option pour activer/d ésactiver l'injection automatiq ue des en-têtes de sécurité HTTP des meilleures pratiques dans toutes les réponses de CloudFront
LogGroupProps ?	`logs.LogGroupProps`	Des accessoires fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour le groupe de journaux CloudWatch Logs.

Propriétés du modèle

Nom	Type	Description
AppiGateway	`api.RestApi`	Renvoie une instance de l'API Gateway API créée par le modèle.
ApigateWayCloudWatchRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle qui active la journalisation des accès à partir de l'API Gateway API vers CloudWatc h.
ApigateWayLogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle auquel les journaux

Nom	Type	Description
		d'accès API REST de API Gateway d'API sont envoyés.
CloudFrontLoggingBucket ?	`s3.Bucket`	Renvoie une instance du compartiment de journalisation créé par le modèle pour la distribution Web CloudFront.
CloudFrontWebDistribution	`cloudfront.CloudFrontWebDistribution`	Renvoie une instance de la distribution Web CloudFront créée par le modèle.
EdgelAmbDAFunction Version ?	`lambda.Version`	Renvoie une instance de la version de la fonction de bord Lambda créée par le motif.
LambdaFunction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Amazon CloudFront

- Configurer la journalisation d'accès pour CloudFront WebDistribution
- Activer l'injection automatique des en-têtes de sécurité HTTP des meilleures pratiques dans toutes les réponses de CloudFront WebDistribution

Amazon API Gateway

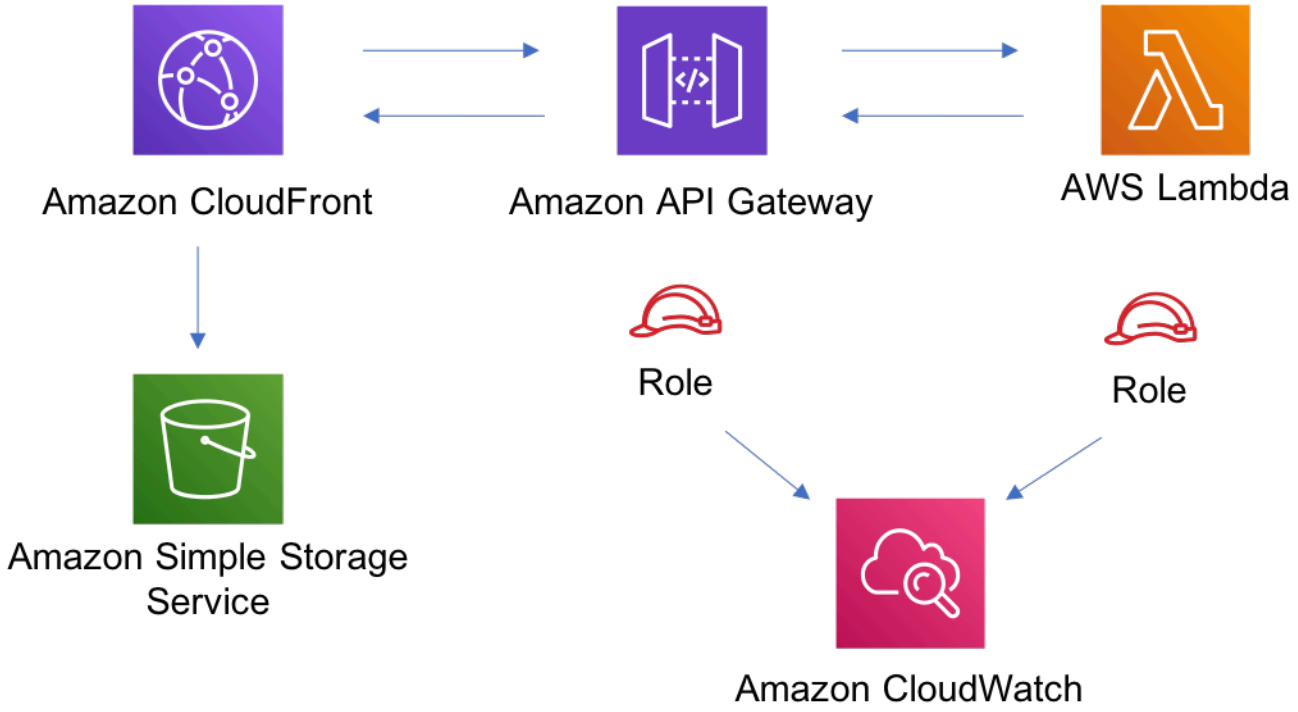
- Déployer un point de terminaison API régional
- Activer la journalisation CloudWatch pour API Gateway
- Configurer le rôle IAM d'accès minimal aux privilèges pour API Gateway

- Définissez l'AuthorizationType par défaut pour toutes les méthodes d'API sur IAM
- Activer le suivi X-Ray

Fonction AWS Lambda

- Configurer le rôle IAM d'accès limité pour la fonction Lambda
- Activer la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda
- Activer le suivi X-Ray
- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

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

aws-cloudfront-mediastore

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion des versions sémantiquesModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Ce module AWS Solutions Construct implémente une distribution Amazon CloudFront connectée à un conteneur AWS Elemental MediaStore.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`CloudFrontToMediaStoreProps`

Accessoires de construction de modèle

Nom	Type	Description
ExistingMediaStoreContainerObj ?	`mediastore.CfnContainer`	Conteneur MediaStore fourni en option par l'utilisateur pour remplacer le conteneur MediaStore par défaut.
MediaStoreContainerProps ?	`mediastore.CfnContainerProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour le conteneur MediaStore.
CloudFrontDistributionProps ?	`cloudfront.DistributionProps` | `any`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour la distribution CloudFront.
InserThttpSecurityHeaders ?	`boolean`	Props fournis par l'utilisateur en option pour activer/désactiver l'injection automatiq

Nom	Type	Description
		ue des en-têtes de sécurité HTTP des meilleures pratiques dans toutes les réponses de CloudFront.

Propriétés du modèle

Nom	Type	Description
CloudFrontWebDistribution	`cloudfront.CloudFrontWebDistribution`	Renvoie une instance de la distribution Web CloudFront créée par le modèle.
MediaStoreContainer	`mediastore.CfnContainer`	Renvoie une instance du conteneur MediaStore créé par le modèle.
CloudFrontLoggingBucket	`s3.Bucket`	Renvoie une instance du compartiment de journalisation créé par le modèle pour la distribution Web CloudFront.
CloudFrontoriginRequestPolicy	`cloudfront.OriginRequestPolicy`	Renvoie une instance de la stratégie de demande d'origine CloudFront créée par le modèle pour la distribution Web CloudFront.
CloudFrontOriginAccessIdentity ?	`cloudfront.OriginAccessIdentity`	Renvoie une instance de l'identité d'accès d'origine CloudFront créée par le modèle pour la distribution Web CloudFront.

Nom	Type	Description
EdgelAmbDAFunctionVersion	`lambda.Version`	Renvoie une instance de la version de la fonction de bord Lambda créée par le motif.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

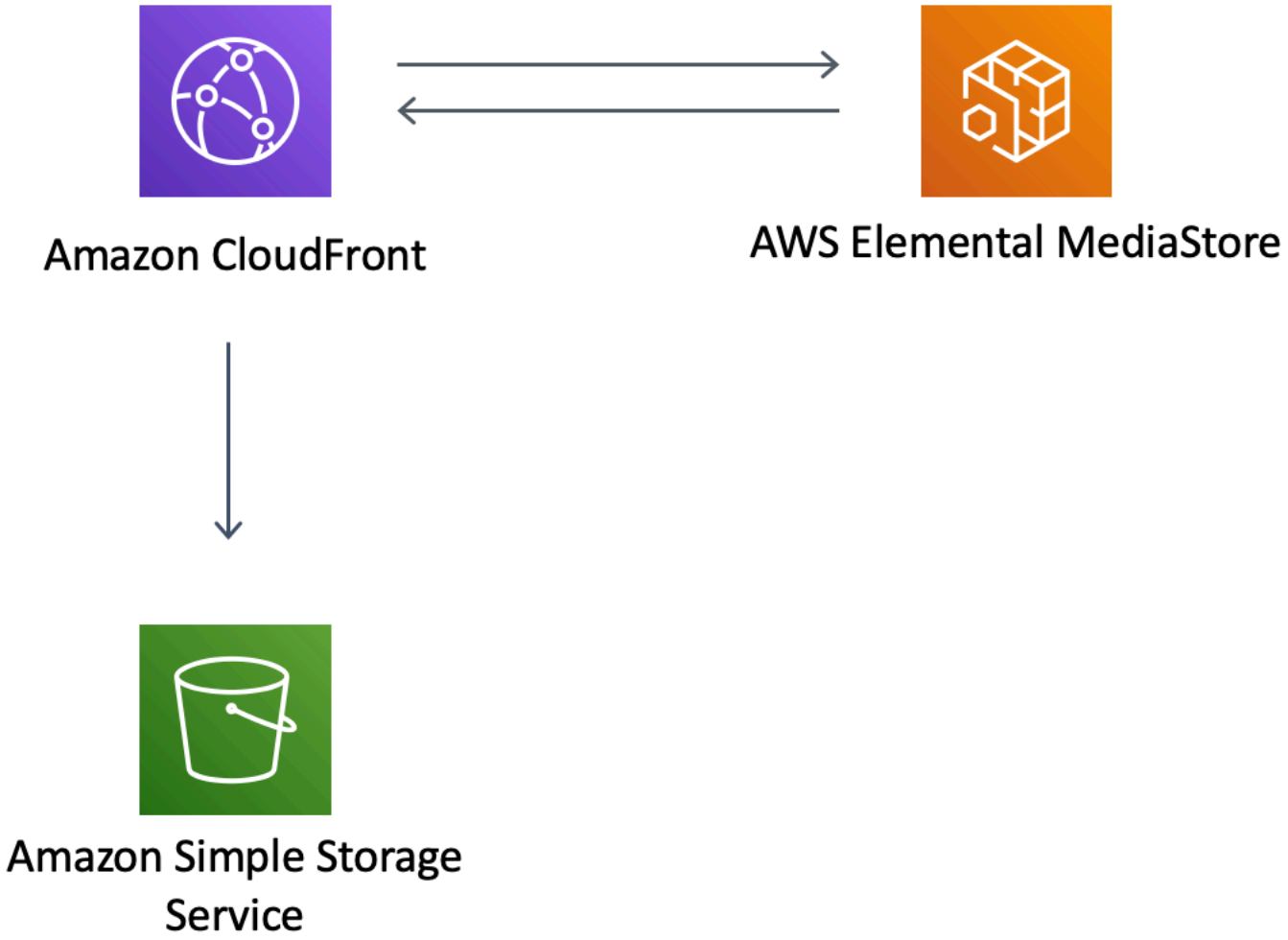
Amazon CloudFront

- Configurer la journalisation des accès pour la distribution Web CloudFront
- Activer la stratégie de demande d'origine CloudFront pour le conteneur AWS Elemental MediaStore
- Définir`User-Agent`en-tête personnalisé avec identité d'accès à l'origine CloudFront
- Activer l'injection automatique des en-têtes de sécurité HTTP des meilleures pratiques dans toutes les réponses de la distribution Web CloudFront

AWS Elemental MediaStore

- Définir la stratégie de suppression pour conserver la ressource
- Définissez le nom du conteneur avec le nom de la pile CloudFormation
- Définir les paramètres par défautStratégie de partage des ressources cross-origin (CORS) de conteneur
- Définir les paramètres par défautstratégie de cycle de vie des objets
- Définir les paramètres par défautstratégie de conteneurpour autoriser uniquement`aws:UserAgent`avec identité d'accès à l'origine CloudFront
- Définir les paramètres par défautstratégie de métriques
- Activer la journalisation des accès

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructs/aws-cloudfront-mediastore

aws-cloudfront-s3

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion sémantique de versionModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Ce composant AWS Solutions Construct implémente une distribution Amazon CloudFront devant un compartiment Amazon S3.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`CloudFrontToS3Props`

Accessoires de construction de modèle

Nom	Type	Description
Bucketobj existant ?	`s3.Bucket`	Instance existante de l'objet S3 Bucket. Si cela est fourni, alors fournir également `bucketProps` est une erreur.
BucketProps ?	`s3.BucketProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du compartiment. Ignoré si `unexistingBucketObj` est fourni.
CloudFrontDistributionProps ?	`cloudfront.DistributionProps`	Des accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut pour la distribution CloudFront.
InserThttpSecurityHeaders ?	`boolean`	Props fournis par l'utilisateur en option pour activer/désactiver l'injection automatique des en-têtes de sécurité HTTP des meilleures

Nom	Type	Description
		pratiques dans toutes les réponses de CloudFront

Propriétés du modèle

Nom	Type	Description
CloudFrontWebDistribution	`cloudfront.CloudFrontWebDistribution`	Renvoie une instance de la distribution Web CloudFront créée par le modèle.
S3Bucket ?	`s3.Bucket`	Renvoie une instance du compartiment S3 créé par le modèle.
S3LoggingBucket ?	`s3.Bucket`	Renvoie une instance du compartiment de journalisation créé par le modèle pour le compartiment S3.
EdgelAmbDAFunction Version ?	`lambda.Version`	Renvoie une instance de la version de la fonction de bord Lambda créée par le motif.
CloudFrontLoggingBucket ?	`s3.Bucket`	Renvoie une instance du compartiment de journalisation créé par le modèle pour la distribution Web CloudFront.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

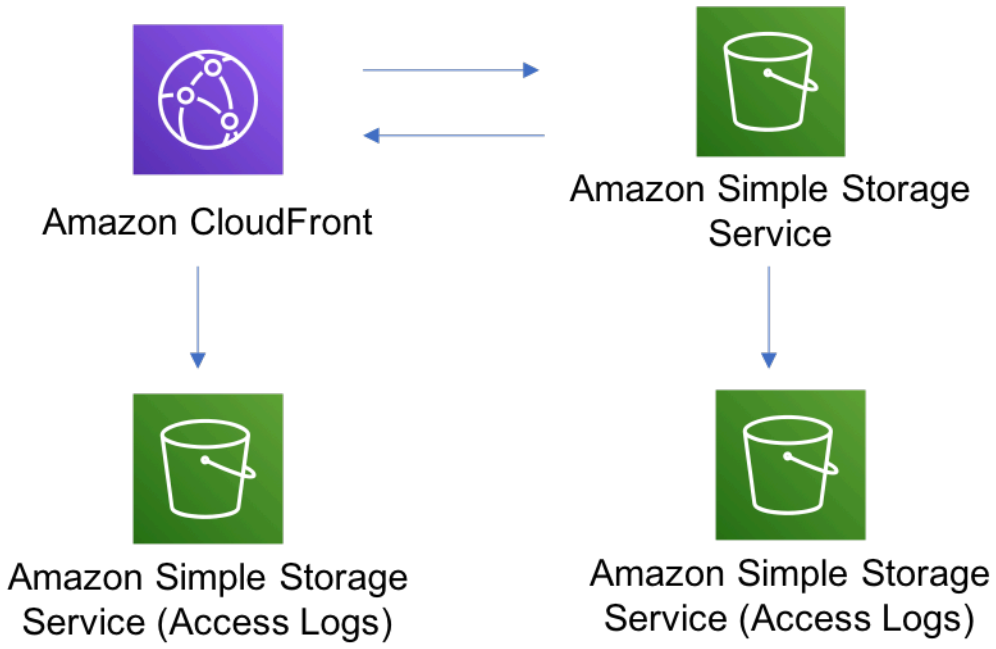
Amazon CloudFront

- Configurer la journalisation d'accès pour CloudFront WebDistribution
- Activer l'injection automatique des en-têtes de sécurité HTTP des meilleures pratiques dans toutes les réponses de CloudFront WebDistribution

Bucket Amazon S3

- Configurer la journalisation d'accès pour le compartiment S3
- Activer le chiffrement côté serveur pour le compartiment S3 à l'aide de la clé KMS gérée par AWS
- Activer le contrôle de version pour S3 Bucket
- Ne pas autoriser l'accès public pour le compartiment S3
- Conserver le compartiment S3 lors de la suppression de la pile CloudFormation
- Application du chiffrement des données en transit
- Applique la règle de cycle de vie pour déplacer les versions d'objets non actuelles vers le stockage Glacier après 90 jours

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-constructs/aws-cloudfront-s3

aws-cognito-apigateway-lambda

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion sémantique des versionsModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Cette solution AWS Construct implémente Amazon Cognito sécurise une API REST basée sur Amazon API Gateway Lambda.

Voici une définition de modèle déployable minimale dans 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'
    }
});
```

Si vous définissez des ressources et des méthodes sur votre API (par exemple`proxy = false`), vous devez appeler le`addAuthorizers()`après que l'API est complètement définie. Cela garantit que toutes les méthodes de votre API sont protégées.

Voici un exemple de 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`CognitoToApiGatewayToLambdaProps`

Props de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.

Nom	Type	Description
ApigatewayProps ?	`api.LambdaRestApiProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour API Gateway
CognitouserPoolProps ?	`cognito.UserPoolProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour le pool d'utilisateurs Cognito
CognitouserPoolClientProps ?	`cognito.UserPoolClientProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour Cognito User Pool Client
LogGroupProps ?	`logs.LogGroupProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour le groupe de journaux CloudWatch Logs.

Propriétés de modèle

Nom	Type	Description
Apigateway	`api.RestApi`	Renvoie une instance de l'API Gateway API créée par le modèle.
LambdaFunction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.

Nom	Type	Description
userPool	`cognito.UserPool`	Renvoie une instance du pool d'utilisateurs Cognito créé par le modèle.
UserPoolClient	`cognito.UserPoolClient`	Renvoie une instance du client de pool d'utilisateurs Cognito créé par le modèle.
ApigateWayCloudWatchRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle qui active la journalisation des accès à partir de l'API Gateway API vers CloudWatch.
ApigateWayLogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle auquel les journaux d'accès API REST de API Gateway d'API sont envoyés.
ApigatewayAuthorizer	`api.CfnAuthorizer`	Renvoie une instance de l'autorisation API Gateway créée par le modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Amazon Cognito

- Définition d'une stratégie de mot de passe pour les groupes
- Appliquer le mode de sécurité avancé pour les pools d'utilisateurs

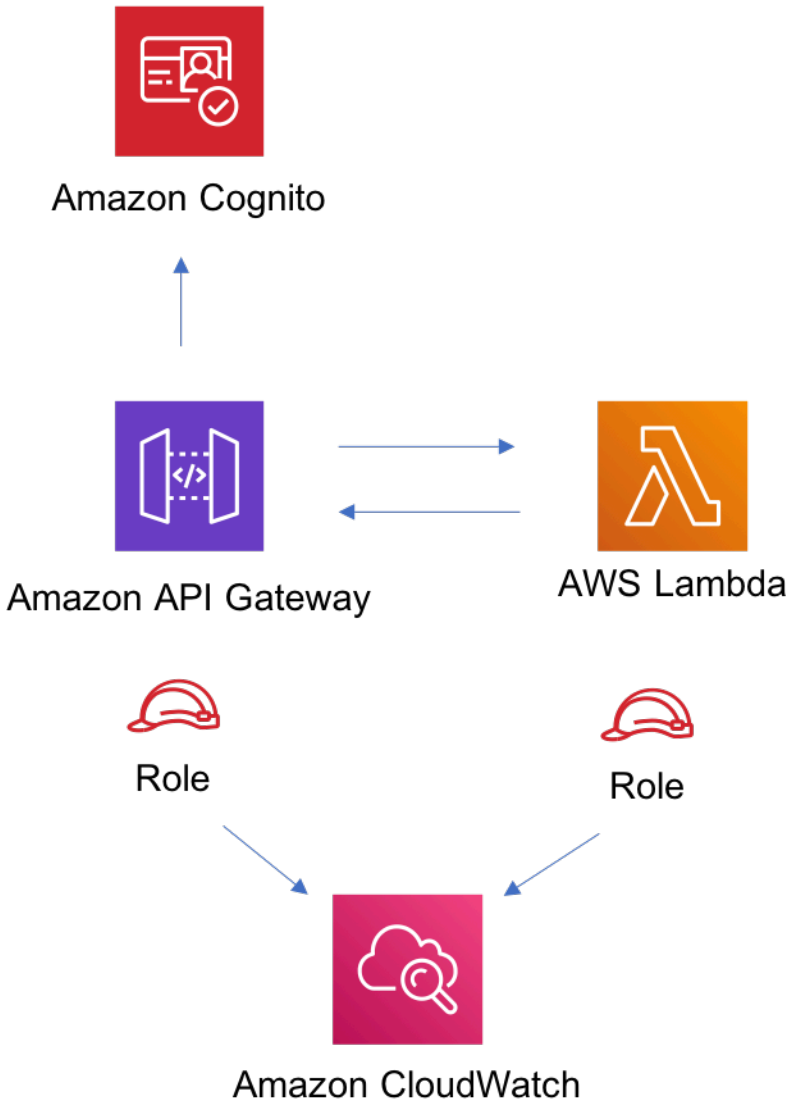
Amazon API Gateway

- Déployer un point de terminaison API optimisé pour les périphériques
- Activer la journalisation CloudWatch pour API Gateway
- Configurer le rôle IAM d'accès minimal aux privilèges pour API Gateway
- Définissez l'AuthorizationType par défaut pour toutes les méthodes d'API sur IAM
- Activer le suivi X-Ray

Fonction AWS Lambda

- Configuration du rôle IAM d'accès limité pour la fonction Lambda
- Activer la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda
- Activer le suivi X-Ray
- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-construction/aws-cognito-apigateway-lambda

aws-dynamodb-stream-lambda

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion sémantique de versionsModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Ce modèle AWS Solutions Construct implémente une table Amazon DynamoDB avec flux pour appeler la fonction AWS Lambda avec les autorisations les moins privilégiées.

Voici une définition de modèle déployable minimale :

```
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);
```

Paramètres

- scope`Construct`
- id`string`
- props`DynamoDBStreamToLambdaProps`

Modèle de construction d'accessoires

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.

Nom	Type	Description
DynamoTableProps ?	`dynamodb.TableProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour DynamoDB Table
ExistantTableObj ?	`dynamodb.Table`	Instance existante de l'objet de table DynamoDB, fournissant à la fois ceci et`dynamoTableProps` provoquera une erreur.
DynamoEventSourceProps ?	`aws-lambda-event-sources.DynamoEventSourceProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour la source d'événements DynamoDB

Propriétés du modèle

Nom	Type	Description
DynamoTable	`dynamodb.Table`	Renvoie une instance de la table DynamoDB créée par le modèle.
LambdaFunction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.

Fonction Lambda

Ce modèle nécessite une fonction Lambda qui peut publier des données dans le service Elasticsearch à partir du flux DynamoDB. Un exemple de fonction est fourni ici.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

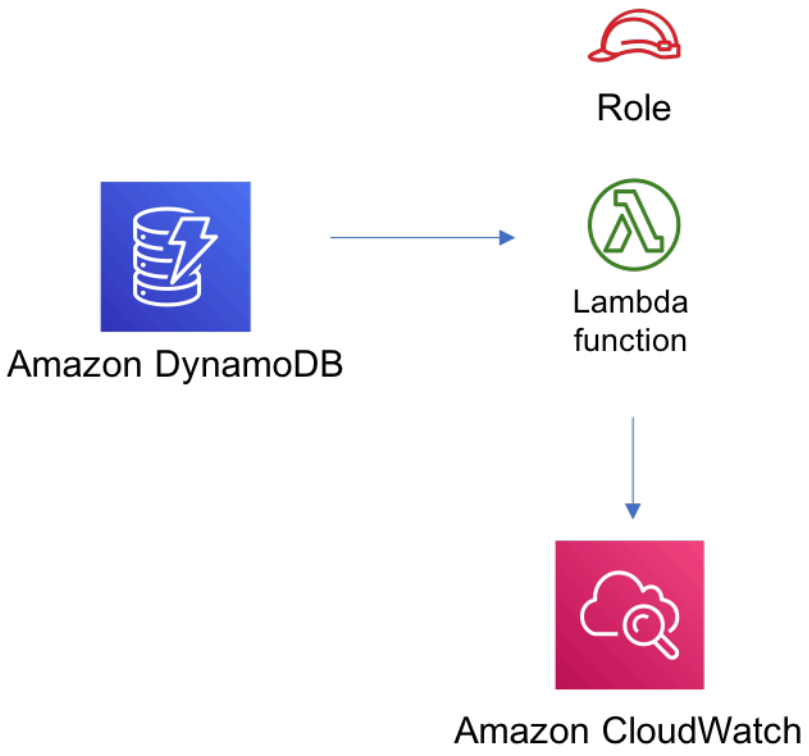
Amazon DynamoDB Table

- Définir le mode de facturation de la table DynamoDB à la demande (Paiement par demande)
- Activer le chiffrement côté serveur pour la table DynamoDB à l'aide de la clé KMS gérée par AWS
- Crée une clé de partition appelée 'id' pour la table DynamoDB
- Conserver la table lors de la suppression de la pile CloudFormation
- Activer les sauvegardes continues et la restauration à un instant dans le passé

Fonction AWS Lambda

- Configuration du rôle IAM d'accès limité pour la fonction Lambda
- Activer la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda
- Activer le suivi X-Ray
- Activer les fonctionnalités de gestion des défaillances : activer bisect sur la fonction Erreur ; définir l'âge maximal des enregistrements par défaut (24 heures) ; définir les tentatives de relance maximales par défaut (500) ; et déployer la file d'attente des lettres mortes SQS comme destination en cas d'échec
- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

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

aws-dynamodb-stream-lambda-elasticsearch-kibana

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Ceux-ci ne sont pas assujettis à laGestion sémantique des versionsModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
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

Cette solution AWS Construct implémente la table Amazon DynamoDB avec flux, une fonction AWS Lambda et un Amazon Elasticsearch Service avec les autorisations les moins privilégiées.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`DynamoDBStreamToLambdaToElasticSearchAndKibanaProps`

Modèle de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.
DynamoTableProps ?	`dynamodb.TableProps`	Props fournis par l'utilisateur en option pour remplacer les

Nom	Type	Description
		accessoires par défaut pour DynamoDB Table
ExistantTableObj ?	`dynamodb.Table`	Instance existante de l'objet de table DynamoDB, fournissant à la fois ceci et `dynamoTableProps` provoquera une erreur.
DynamoEventSourceProps ?	`aws-lambda-event-sources.DynamoEventSourceProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour la source d'événements DynamoDB
ESDomainProps ?	`elasticsearch.CfnDomainProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour Amazon Elasticsearch Service
domainName	`string`	Nom de domaine pour Cognito et Amazon Elasticsearch Service
CreateCloudWatchArms	`boolean`	Indique s'il faut créer des alarmes CloudWatch recommandées.

Propriétés du modèle

Nom	Type	Description
Cloudwatch Alarm ?	`cloudwatch.Alarm[]`	Renvoie la liste d'une ou plusieurs alarmes CloudWatch créées par le modèle.

Nom	Type	Description
DynamoTable	`dynamodb.Table`	Renvoie une instance de la table DynamoDB créée par le modèle.
ElasticSearchDomain	`elasticsearch.CfnDomain`	Renvoie une instance du domaine Elasticsearch créé par le modèle.
IdentityPool	`cognito.CfnIdentityPool`	Renvoie une instance du pool d'identités Cognito créé par le modèle.
LambdaFunction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.
userPool	`cognito.UserPool`	Renvoie une instance du pool d'utilisateurs Cognito créé par le modèle.
UserPoolClient	`cognito.UserPoolClient`	Renvoie une instance du client de pool d'utilisateurs Cognito créé par le modèle.

Fonction Lambda

Ce modèle nécessite une fonction Lambda qui peut publier des données dans le service Elasticsearch à partir du flux DynamoDB. Un exemple de fonction est fourniIci.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Amazon DynamoDB Table

- Définir le mode de facturation de la table DynamoDB à la demande (Paiement par demande)

- Activer le chiffrement côté serveur pour la table DynamoDB à l'aide de la clé KMS gérée par AWS
- Crée une clé de partition appelée 'id' pour la table DynamoDB
- Conserver la table lors de la suppression de la pile CloudFormation
- Activer les sauvegardes continues et la restauration à un instant dans le passé

Fonction AWS Lambda

- Configuration d'un rôle IAM à accès limité pour la fonction Lambda
- Activer la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda
- Activer le suivi X-Ray
- Activer les fonctionnalités de gestion des défaillances : activer bisect sur la fonction Erreur ; définir l'âge maximal des enregistrements par défaut (24 heures) ; définir les tentatives de relance maximales par défaut (500) ; et déployer la file d'attente des lettres mortes SQS comme destination en cas d'échec
- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

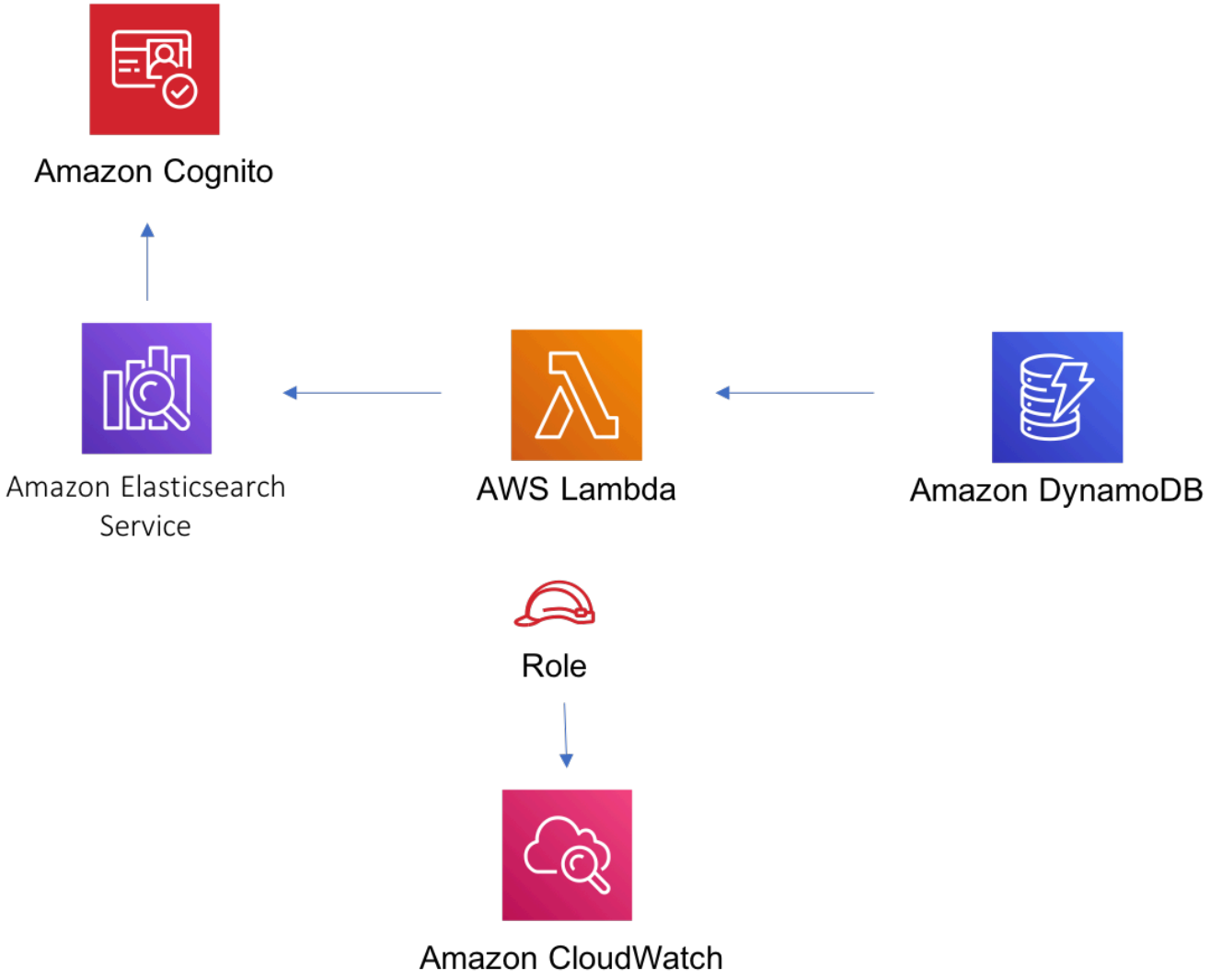
Amazon Cognito

- Définir la stratégie de mot de passe des groupes d'utilisateurs
- Appliquer le mode de sécurité avancé pour les pools d'utilisateurs

Amazon Elasticsearch Service

- Déployer les meilleures pratiques des alarmes CloudWatch pour le domaine Elasticsearch
- Sécurisez l'accès au tableau de bord Kibana avec Cognito User Pools
- Activer le chiffrement côté serveur pour le domaine Elasticsearch à l'aide de la clé KMS gérée par AWS
- Activer le chiffrement nœud à nœud pour le domaine Elasticsearch
- Configuration du cluster pour le domaine Amazon ES

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	[@aws -solutions-constructs/aws-dynamodb-stream-lambda-elasticsearch-kibana]

aws-evenements-rule-kinesisfirehose-s3

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion sémantiqueLe modèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
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

AWS Solutions Construct implémente une règle Amazon CloudWatch Events pour envoyer des données à un flux de distribution Amazon Kinesis Data Firehose connecté à un compartiment Amazon S3.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`EventsRuleToKinesisFirehoseToS3Props`

Accessoires de construction de modèle

Nom	Type	Description
EventTruleProps	`events.RuleProps`	Propriétés fournies par l'utilisateur pour remplacer les propriétés par défaut de la règle CloudWatch Events.
KinesisFireHoseProps ?	`aws-kinesisfirehose.CfnDeliveryStreamProps`	Des accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut pour Kinesis Firehose Delivery Stream.
Bucketobj existant ?	`s3.IBucket`	Instance existante de l'objet S3 Bucket. Si cela est fourni,

Nom	Type	Description
		alors fournir également `bucketProps` est une erreur.
BucketProps ?	`s3.BucketProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour le compartiment S3.
LogGroupProps ?	`logs.LogGroupProps`	Options fournies par l'utilisa teur pour remplacer les accessoires par défaut pour le groupe de journaux CloudWatch Logs.

Propriétés de modèle

Nom	Type	Description
EventsRègle	`events.Rule`	Renvoie une instance de la règle Events créée par le modèle.
KinesisFireHose	`kinesisfirehose.CfnDeliveryStream`	Renvoie une instance du flux de livraison Kinesis Firehose créé par le modèle.
S3Bucket	`s3.Bucket`	Renvoie une instance du compartiment S3 créé par le modèle.
S3LoggingBucket ?	`s3.Bucket`	Renvoie une instance du compartiment de journalisation créé par le modèle pour le compartiment S3.

Nom	Type	Description
EventsRole ?	`iam.Role`	Renvoie une instance du rôle créé par la construction pour la règle CloudWatch Events.
KineSisFireHoserole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle pour le flux de livraison Kinesis Firehose.
KineSisFireHoselogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle auquel les journaux d'accès Kinesis Firehose sont envoyés.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Règle Amazon CloudWatch Events

- Configurez le rôle IAM d'accès minimal aux privilèges pour que la règle des événements soit publiée dans le flux de distribution Kinesis Firehose.

Amazon Kinesis Firehose

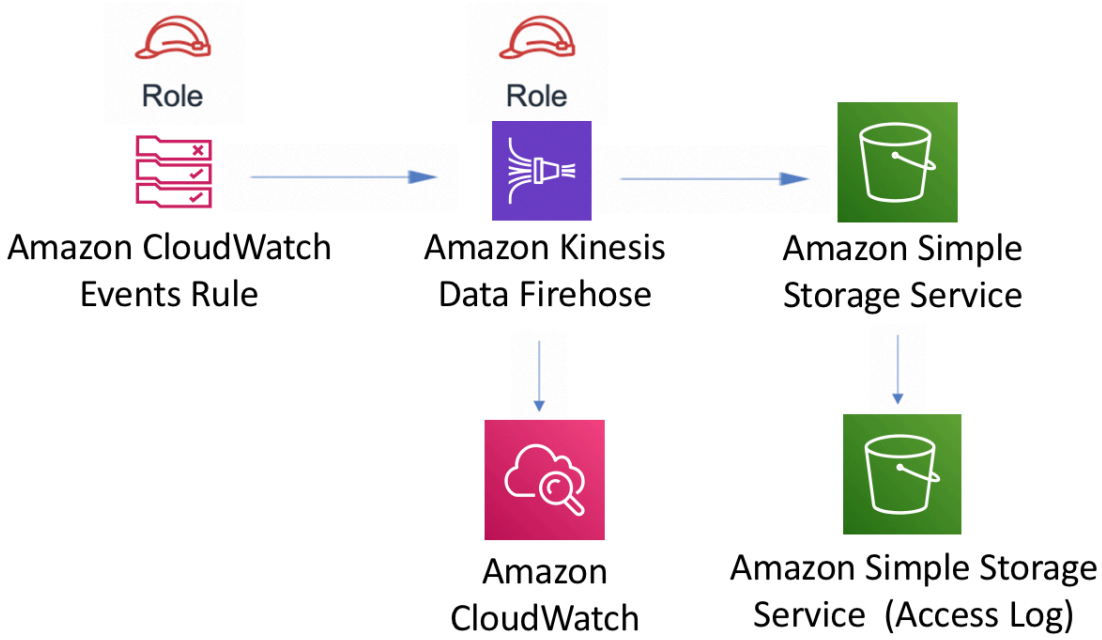
- Activez la journalisation CloudWatch pour Kinesis Firehose.
- Paramètre le rôle IAM pour Amazon Kinesis Firehose.

Compartiment Amazon S3

- Configurez la journalisation d'accès pour le compartiment.
- Activez le chiffrement côté serveur pour le compartiment à l'aide de la clé KMS gérée par AWS.
- Activer la gestion de version pour le compartiment.

- N'autorisez pas l'accès public au compartiment.
- Conservez le compartiment lors de la suppression de la pile CloudFormation.
- Applique la règle de cycle de vie pour déplacer les versions d'objets non actuelles vers le stockage Glacier après 90 jours.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

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

aws-events-rule-kinesisstreams

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Ceux-ci ne sont pas assujettis à laGestion sémantique de versionModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Cette solution AWS Construct implémente une règle Amazon CloudWatch Events pour envoyer des données à un flux de données Amazon Kinesis.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`EventsRuleToKinesisStreamsProps`

Modèle de construction d'accessoires

Nom	Type	Description
EventTruleProps	`events.RuleProps`	Propriétés fournies par l'utilisateur pour remplacer les propriétés par défaut de la règle CloudWatch Events.
L'existence de Streamobj ?	`kinesis.Stream`	Instance existante de Kinesis Stream, fournissant à la fois ceci et`kinesisStreamProps` provoquera une erreur.
KinesisStreamProps ?	`kinesis.StreamProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour le flux Kinesis.

Nom	Type	Description
CreateCloudWatchArms	`boolean`	Indique s'il faut créer des alarmes CloudWatch recommandées.

Propriétés de modèle

Nom	Type	Description
EventsRègle	`events.Rule`	Renvoie une instance de la règle Events créée par le modèle.
Stream Kinesis	`kinesis.Stream`	Renvoie une instance du flux Kinesis créé par le modèle.
EventsRole ?	`iam.Role`	Renvoie une instance du rôle créé par la construction pour la règle CloudWatch Events.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Amazon CloudWatch Events

- Configurez le rôle IAM d'accès minimal aux privilèges pour que la règle des événements soit publiée dans le flux de données Kinesis.

Amazon Kinesis Stream

- Activez le chiffrement côté serveur pour Kinesis Data Stream à l'aide de la clé KMS gérée AWS.

Architecture

GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

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

aws-événements-rule-lambda

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion sémantiqueModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-

être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Ce modèle AWS Solutions Construct implémente une règle AWS Events et une fonction AWS Lambda.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`EventsRuleToLambdaProps`

Modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.
EventTruleProps	`events.RuleProps`	L'utilisateur a fourni EvenTrule Props pour remplacer les valeurs par défaut

Propriétés de modèle

Nom	Type	Description
EventsRègle	`events.Rule`	Renvoie une instance de la règle Events créée par le modèle.
LambdaFunction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

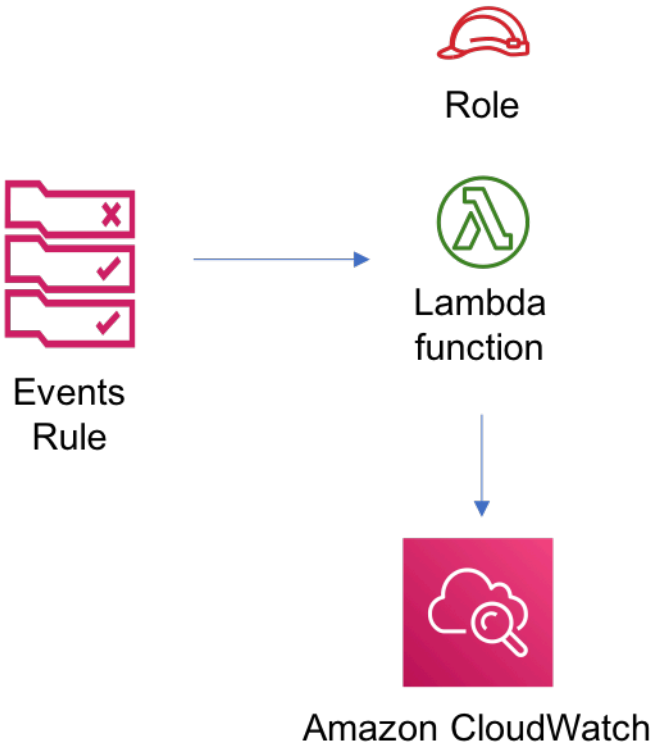
Amazon CloudWatch Events

- Accorder les autorisations les moins privilèges aux événements CloudWatch pour déclencher la fonction Lambda

Fonction AWS Lambda

- Configuration du rôle IAM d'accès limité pour la fonction Lambda
- Activer la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda
- Activer le suivi X-Ray
- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructions/aws-evenements-rule-lambda

aws-événements-rule-sns

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion de version sémantiqueModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Ce modèle implémente une règle Amazon CloudWatch Events connectée à une rubrique Amazon SNS.

Voici une définition de modèle déployable minimale :

```
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);
```

Paramètres

- scope`Construct`
- id`string`
- props`EventsRuleToSnsProps`

Modèle de construction d'accessoires

Nom	Type	Description
EventTruleProps	`events.RuleProps`	Propriétés fournies par l'utilisateur pour remplacer les propriétés par défaut de la règle CloudWatch Events.
ExistantTopicObj ?	`sns.Topic`	Instance existante de l'objet SNS Topic, fournissant à la fois ceci et`topicProps` provoquera une erreur.
SujetProps ?	`sns.TopicProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de

Nom	Type	Description
		la rubrique SNS. Ignoré si une`existingTopicObj` est fourni.
EnableEncryptionWithCustomerManagedKey ?	`boolean`	Indique s'il faut utiliser une clé de chiffrement gérée par le client, soit gérée par cette application CDK, soit importée. Si vous importez une clé de chiffrement, elle doit être spécifiée dans la`encryptionKey` pour cette construction.
encryptionKey ?	`kms.Key`	Une clé de chiffrement existante facultative à utiliser à la place de la clé de chiffrement par défaut.
EncryptionKeyProps ?	`kms.KeyProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la clé de chiffrement.

Propriétés du modèle

Nom	Type	Description
EventsRègle	`events.Rule`	Renvoie une instance de la règle Events créée par le modèle.
snsTopic	`sns.Topic`	Renvoie une instance de la rubrique SNS créée par le modèle.

Nom	Type	Description
encryptionKey	`kms.Key`	Renvoie une instance de la clé de chiffrement créée par le modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

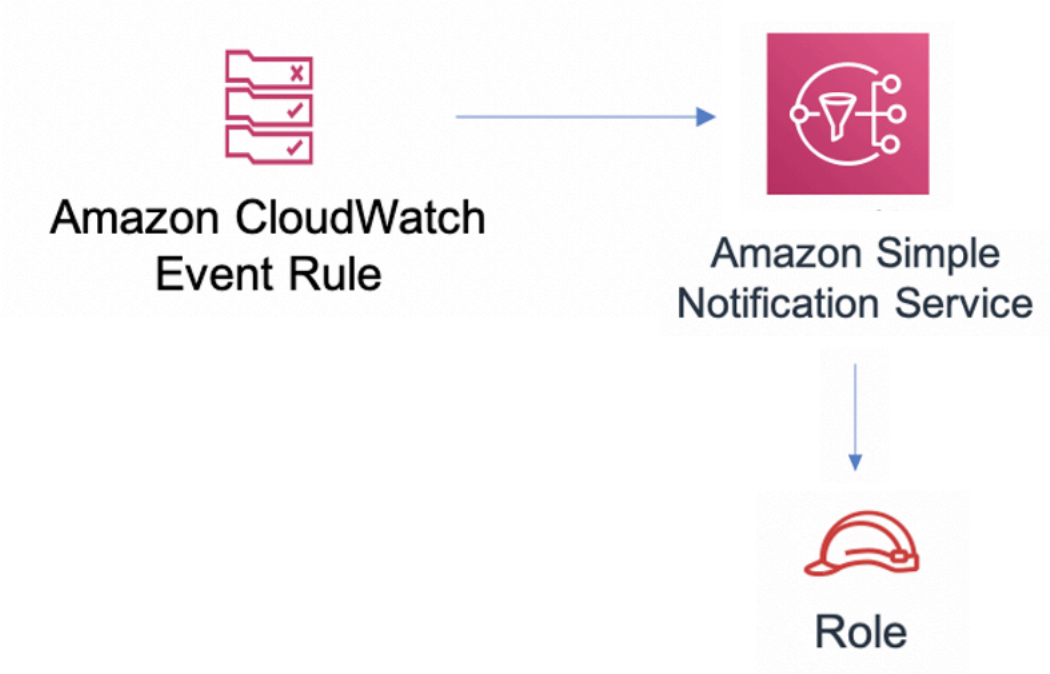
Amazon CloudWatch Events règle

- Accordez les autorisations les moins privilèges aux événements CloudWatch afin de les publier dans la rubrique SNS.

Rubrique Amazon SNS

- Configurez les autorisations d'accès les moins privilèges pour la rubrique SNS.
- Activer le chiffrement côté serveur pour la rubrique SNS à l'aide de la clé AWS KMS gérée par le client.
- Application du chiffrement des données en transit.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

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

aws-evenements-rule-sqs

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas

assujetties à laGestion de versions sémantiquesModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Ce modèle implémente une règle Amazon CloudWatch Events connectée à une file d'attente Amazon SQS.

Voici une définition de modèle déployable minimale :

```
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);
```

Paramètres

- scopeConstruct
- idstring
- propsEventsRuleToSqsProps

Modèle de construction

Nom	Type	Description
EventTruleProps	events.RuleProps	Propriétés fournies par l'utilisa teur pour remplacer les propriétés par défaut de la règle CloudWatch Events.
QueueObj existant ?	sqs.Queue	Une file d'attente SQS existante facultative à utiliser à la place de la file d'attente par défaut. Fournissant à

Nom	Type	Description
		la fois ceci et`queueProps` provoquera une erreur.
QueueProps ?	`sqs.QueueProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la file d'attente SQS. Ignoré si un`existingQueueObj` est fourni.
EnableQueueUrging ?	`boolean`	Indique s'il faut accorder des autorisations supplémentaires à la fonction Lambda lui permettant de purger la file d'attente SQS. La valeur par défaut est `false`.
Déploiement DeadletterQueue ?	`boolean`	Indique s'il faut créer une file d'attente secondaire à utiliser comme file d'attente de lettres mortes. La valeur par défaut est `true`.
DeadletterQueueProps ?	`sqs.QueueProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de la file d'attente de lettres mortes. Utilisé uniquement si le`deployDeadLetterQueue` est défini sur true.

Nom	Type	Description
MaxReceiveCount ?	`number`	Nombre de fois qu'un message peut être déplacé sans succès avant d'être déplacé vers la file d'attente de lettres mortes. La valeur par défaut est 15.
EnableEncryptionWithCustomerManagedKey ?	`boolean`	Indique s'il faut utiliser une clé de chiffrement gérée par le client, soit gérée par cette application CDK, soit importée. Si vous importez une clé de chiffrement, elle doit être spécifiée dans la`encryptionKey` pour cette construction.
encryptionKey ?	`kms.Key`	Une clé de chiffrement existante facultative à utiliser à la place de la clé de chiffrement par défaut.
EncryptionKeyProps ?	`kms.KeyProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la clé de chiffrement.

Propriétés de modèle

Nom	Type	Description
EventsRègle	`events.Rule`	Renvoie une instance de la règle Events créée par le modèle.

Nom	Type	Description
SQSqueue	`sqs.Queue`	Renvoie une instance de la file d'attente SQS créée par le modèle.
encryptionKey	`kms.Key`	Renvoie une instance de la clé de chiffrement créée par le modèle.
DeadletterQueue ?	`sqs.Queue`	Renvoie une instance de la file d'attente de lettres mortes créée par le modèle, si une instance est déployée.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

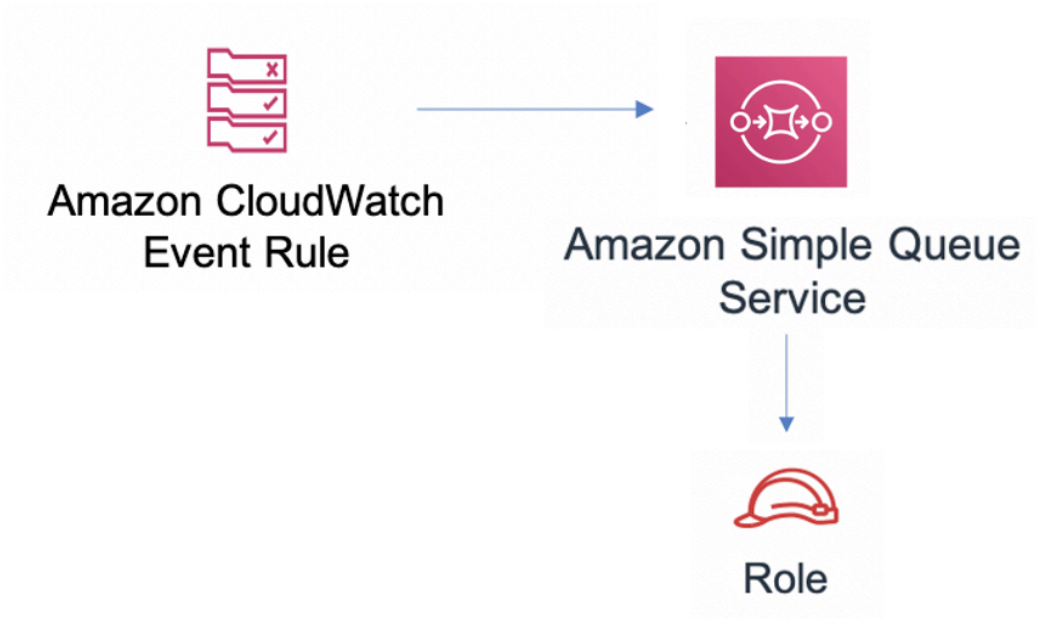
Amazon CloudWatch Events

- Accordez les autorisations les moins privilèges aux événements CloudWatch afin de les publier dans la file d'attente SQS.

File d'attente Amazon SQS

- Déployer une file d'attente de lettres mortes pour la file d'attente source.
- Activer le chiffrement côté serveur pour la file d'attente source à l'aide d'une clé AWS KMS gérée par le client.
- Application du chiffrement des données en transit.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

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

aws-events-rule-step-function

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas

assujetties à laGestion de version sémantiquemodèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_events_rule_step_function`
Typecript	`@aws-solutions-constructs/aws-events-rule-step-function`
Java	`software.amazon.awsconstructs.services.eventsrulestepfunction`

Overview

Cette solution AWS Solutions Construct implémente une règle AWS Events et une fonction AWS Step.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`EventsRuleToStepFunctionProps`

Accessoires de construction de modèle

Nom	Type	Description
StateMachineProps	`sfn.StateMachineProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour SFN.StateMachine
EventTruleProps	`events.RuleProps`	L'utilisateur a fourni EventRule Props pour remplacer les valeurs par défaut
CreateCloudWatchArms	`boolean`	Indique s'il faut créer des alarmes CloudWatch recommandées.
LogGroupProps ?	`logs.LogGroupProps`	Props fournis par l'utilisateur pour remplacer les accessoir

Nom	Type	Description
		es par défaut pour le groupe de journaux CloudWatch Logs.

Propriétés de modèle

Nom	Type	Description
CloudwatAlarm ?	`cloudwatch.Alarm[]`	Renvoie une liste d'une ou plusieurs alarmes CloudWatch créées par le modèle.
EventsRègle	`events.Rule`	Renvoie une instance de la règle Events créée par le modèle.
StateMachine	`sfn.StateMachine`	Renvoie une instance de la machine d'état créée par le modèle.
StateMachineLogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle pour la machine d'état.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

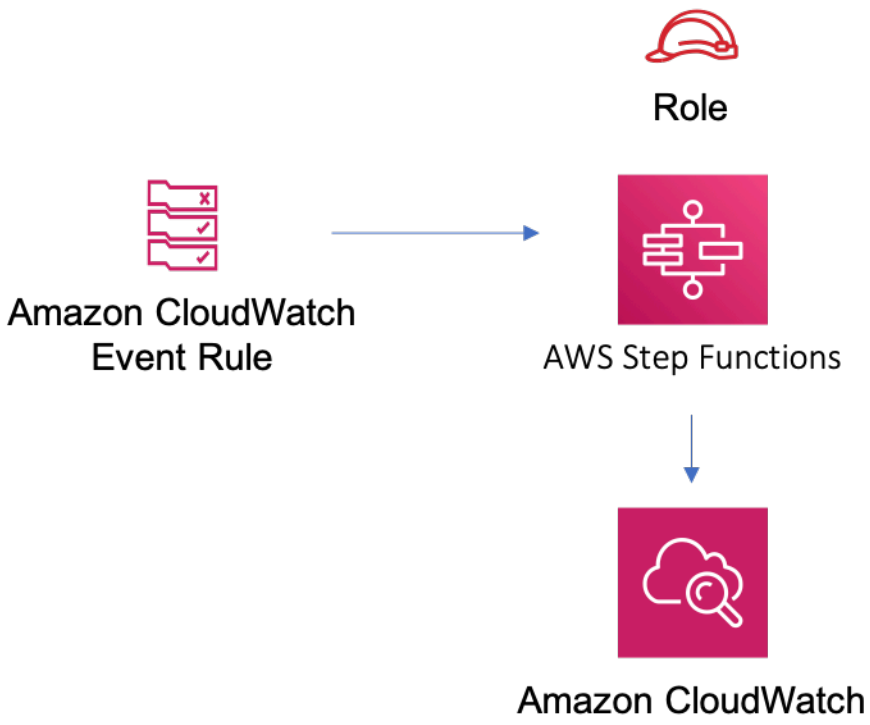
Amazon CloudWatch Events

- Accorder les autorisations les moins privilèges aux événements CloudWatch pour déclencher la fonction Lambda

AWS Step Functions

- Activer la journalisation CloudWatch pour API Gateway
- Déployer les meilleures pratiques des alarmes CloudWatch pour la fonction Step

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructs/aws-events-rule-step-function

aws-iot-kinesisfirehose-s3

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Ceux-ci ne sont pas assujettis à laGestion de version sémantiqueModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_iot_kinesisfirehose_s3`
Typecript	`@aws-solutions-constructs/aws-iot-kinesisfirehose-s3`
Java	`software.amazon.awsconstructs.services.iotkinesisfirehoses3`

Overview

Cette solution AWS Solutions Construct implémente une règle de rubrique AWS IoT MQTT pour envoyer des données à un flux de distribution Amazon Kinesis Data Firehose connecté à un compartiment Amazon S3.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`IotToKinesisFirehoseToS3Props`

Accessoires de construction de modèle

Nom	Type	Description
IotTopicRuleProps	`iot.CfnTopicRuleProps`	L'utilisateur a fourni CFNTopicRuleProps pour remplacer les valeurs par défaut
KinesisFireHoseProps ?	`kinesisfirehose.CfnDeliveryStreamProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour Kinesis Firehose Delivery Stream

Nom	Type	Description
Bucketobj existant ?	`s3.Bucket`	Instance existante de l'objet S3 Bucket, fournissant à la fois ceci et`bucketProps` entraînera une erreur.
BucketProps ?	`s3.BucketProps`	L'utilisateur a fourni des accessoires pour remplacer les accessoires par défaut pour le compartiment S3. Si cela est fourni, alors fournir également`bucketProps` est une erreur.
LogGroupProps ?	`logs.LogGroupProps`	Accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut du groupe de journaux CloudWatch Logs.

Propriétés du modèle

Nom	Type	Description
IOtActionsRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle pour la règle IoT.
IotTopicRule	`iot.CfnTopicRule`	Renvoie une instance de la règle de rubrique IoT créée par le modèle.
KinesisFireHose	`kinesisfirehose.CfnDeliveryStream`	Renvoie une instance du flux de livraison Kinesis Firehose créé par le modèle.

Nom	Type	Description
KineSisFireHoselogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle auquel les journaux d'accès Kinesis Firehose sont envoyés.
KineSisFireHoserole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle pour le flux de livraison Kinesis Firehose.
S3Bucket ?	`s3.Bucket`	Renvoie une instance du compartiment S3 créé par le modèle.
S3LoggingBucket ?	`s3.Bucket`	Renvoie une instance du compartiment de journalisation créé par le modèle pour le compartiment S3.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Règle Amazon IoT

- Configurer le rôle IAM d'accès minimal aux privilèges pour Amazon IoT

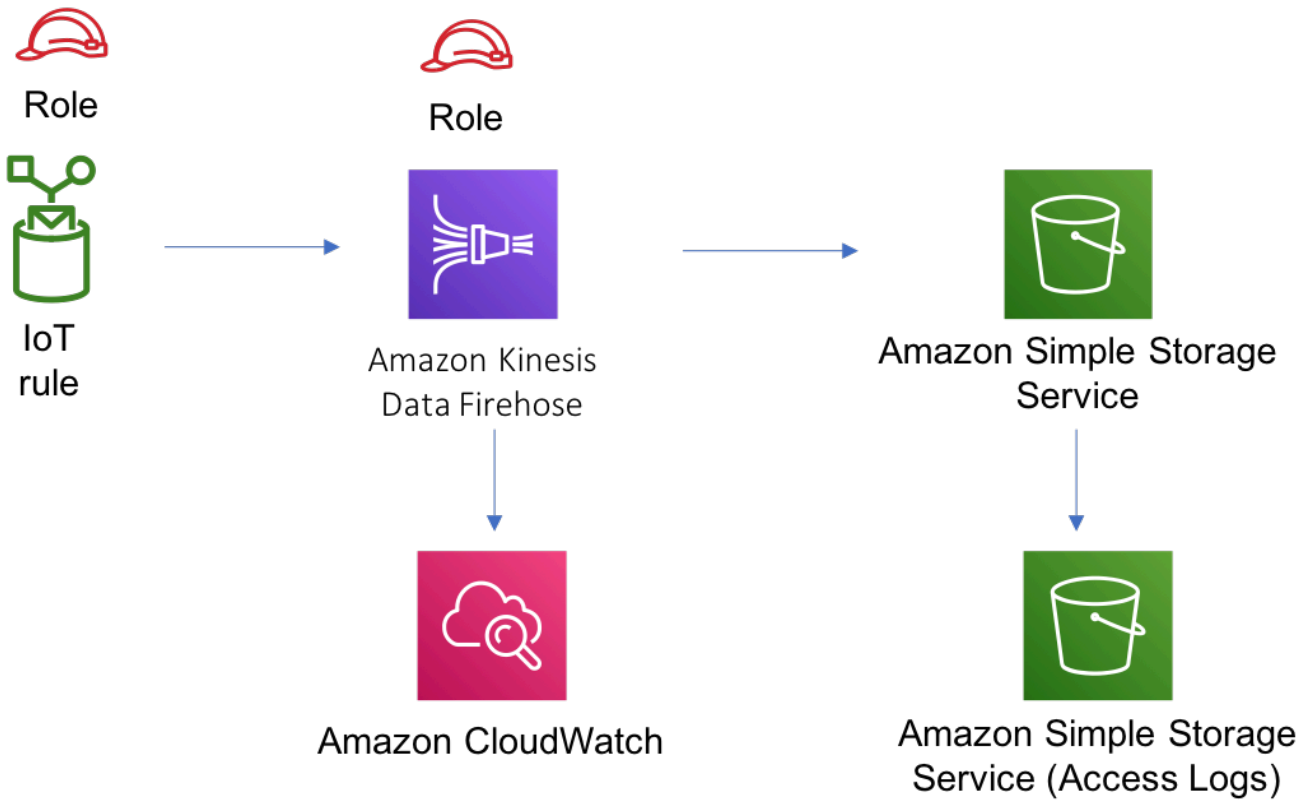
Amazon Kinesis Firehose

- Activer la journalisation CloudWatch pour Kinesis Firehose
- Configurer le rôle IAM d'accès minimal aux privilèges pour Amazon Kinesis Firehose

Bucket Amazon S3

- Configurer la journalisation d'accès pour le compartiment S3
- Activer le chiffrement côté serveur pour le compartiment S3 à l'aide de la clé KMS gérée par AWS
- Activer le contrôle de version pour S3 Bucket
- Ne pas autoriser l'accès public pour le compartiment S3
- Conserver le compartiment S3 lors de la suppression de la pile CloudFormation
- Applique la règle de cycle de vie pour déplacer les versions d'objets non actuelles vers le stockage Glacier après 90 jours

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-constructs/aws-iot-kinesisfirehose-s3

aws-iot-lambda

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Ceux-ci ne sont pas assujettis à laGestion sémantique de versionsModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Ce modèle AWS Solutions Constructs implémente une règle de rubrique AWS IoT MQTT et un modèle de fonction AWS Lambda.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`

- `idstring`
- propsIotToLambdaProps

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	lambda.Function	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` entraînera une erreur.
LambdaFunctionProps ?	lambda.FunctionProps	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.
IotopicRuleProps ?	iot.CfnTopicRuleProps	L'utilisateur a fourni CFNTopicRuleProps pour remplacer les valeurs par défaut

Propriétés de modèle

Nom	Type	Description
IotTopicRule	iot.CfnTopicRule	Renvoie une instance de la règle de rubrique IoT créée par le modèle.
LambdaUnction	lambda.Function	Renvoie une instance de la fonction Lambda créée par le modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

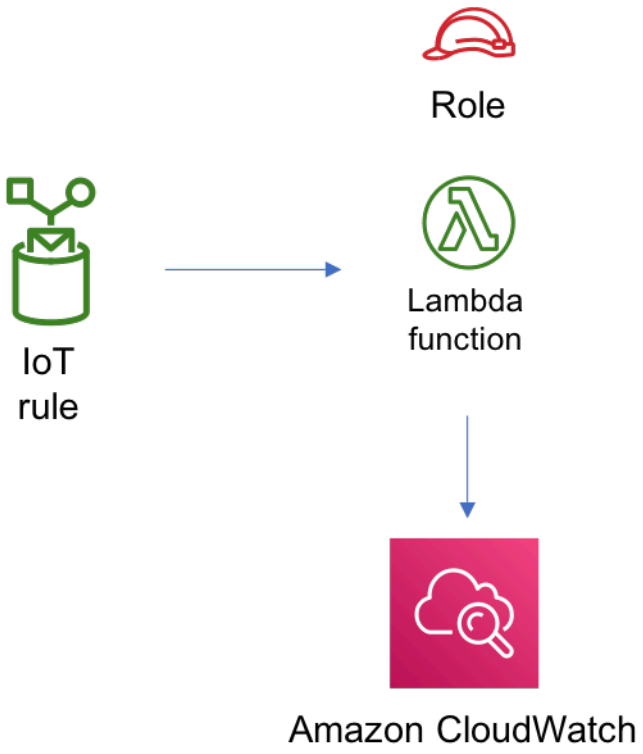
Règle Amazon IoT

- Configurez le rôle IAM d'accès le moins élevé pour Amazon IoT.

Fonction AWS Lambda

- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Activer le suivi X-Ray.
- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-constructions/aws-iot-lambda

aws-iot-lambda-dynamodb

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion sémantiqueModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_iot_lambda_dynamodb`
Typecript	`@aws-solutions-constructs/aws-iot-lambda-dynamodb`
Java	`software.amazon.awsconstructs.services.iotlambdadynamodb`

Overview

Ce modèle AWS Solutions Constructs implémente une règle de rubrique AWS IoT, une fonction AWS Lambda et une table Amazon DynamoDB avec les autorisations les moins privilégiées.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`

- `idstring`
- propsIotToLambdaToDynamoDBProps

Modèle de construction

Nom	Type	Description
L'existence de Glambdaobj ?	lambda.Function	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps	lambda.FunctionProps	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.
IotTopicRuleProps	iot.CfnTopicRuleProps	Les accessoires fournis par l'utilisateur pour remplacer les accessoires par défaut
DynamoTableProps ?	dynamodb.TableProps	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour DynamoDB Table
TableAutorisations ?	string	Autorisations de table facultatives à accorder à la fonction Lambda. L'une des options suivantes peut être spécifiée :`All`,`Read`,`ReadWrite`, ou`Write`.

Propriétés de modèle

Nom	Type	Description
DynamoTable	`dynamodb.Table`	Renvoie une instance de la table DynamoDB créée par le modèle.
IotTopicRule	`iot.CfnTopicRule`	Renvoie une instance de la règle de rubrique IoT créée par le modèle.
LambdaUnction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Amazon IoT Règle

- Configurez le rôle IAM d'accès le moins élevé pour Amazon IoT.

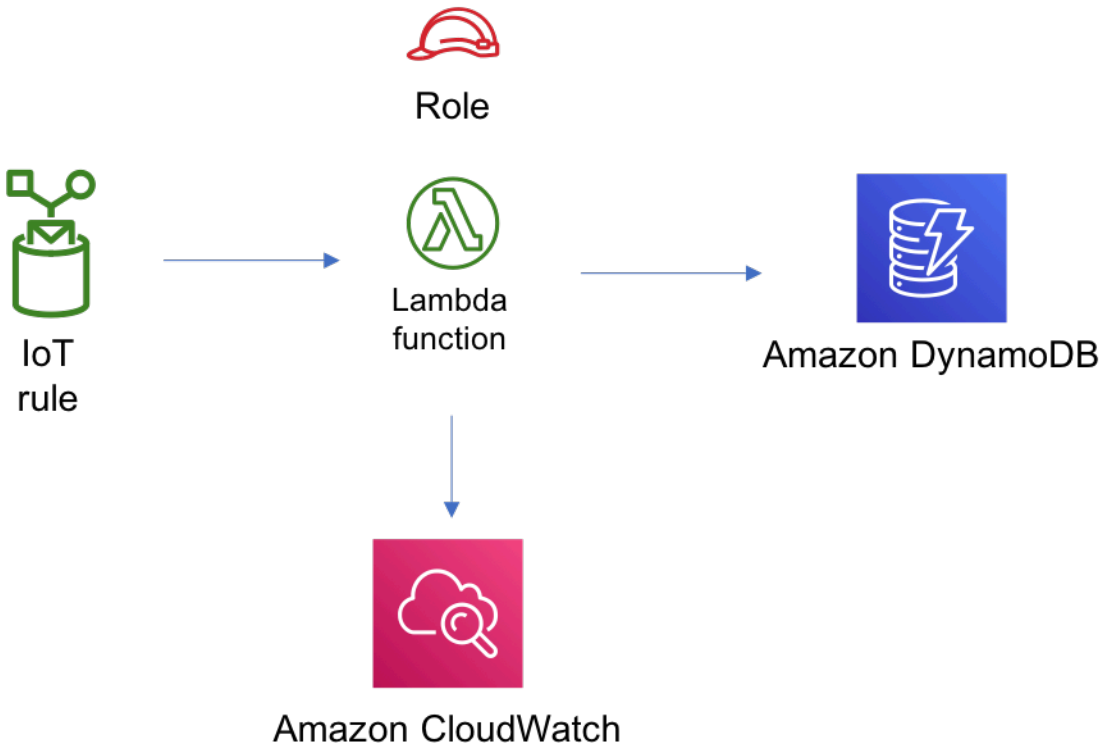
Fonction AWS Lambda

- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Activer le suivi X-Ray.
- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Amazon DynamoDB Table

- Définissez le mode de facturation de la table DynamoDB sur On-Demand (Paiement par demande).
- Activez le chiffrement côté serveur pour la table DynamoDB à l'aide de la clé KMS gérée par AWS.
- Crée une clé de partition appelée 'id' pour DynamoDB Table.
- Conservez la table lors de la suppression de la pile CloudFormation.
- Permet de procéder à des sauvegardes continues et à une restauration à un instant dans le passé.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

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

aws-kinesisfirehose-s3

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion de version sémantiquemodèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Cette solution AWS Solutions Construct implémente un flux de diffusion Amazon Kinesis Data Firehose connecté à un compartiment Amazon S3.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scopeConstruct
- idstring
- propsKinesisFirehoseToS3Props

Accessoires de construction de modèle

Nom	Type	Description
BucketProps ?	s3.BucketProps	Des accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut pour le compartiment S3.
Bucketobj existant ?	s3.IBucket	Instance existante facultative de S3 Bucket. Si cela est fourni, alors fournir également bucketProps est une erreur.
ExistingLoggingBucketObj ?	s3.IBucket	Instance existante facultative de journalisation du compartiment S3 pour le compartiment S3 créé par le modèle.
KinesisFireHoseProps ?	kinesisfirehose.CfnDeliveryStreamProps | any	Des accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut pour Kinesis Firehose Delivery Stream.

Nom	Type	Description
LogGroupProps ?	`logs.LogGroupProps`	Les accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut pour CloudWatchLogs LogGroup.

Propriétés du modèle

Nom	Type	Description
KinesisFireHose	`kinesisfirehose.CfnDeliveryStream`	Renvoie une instance de KinesisFireHose.cfnDelivery Stream créée par la construction.
KineSisFireHoselogGroup	`logs.LogGroup`	Renvoie une instance du logs.logGroup créé par la construction pour le flux de diffusion Kinesis Data Firehose.
KineSisFireHoserole	`iam.Role`	Renvoie une instance de l'IAM.Role créée par la construction pour le flux de livraison Kinesis Data Firehose.
S3Bucket ?	`s3.Bucket`	Renvoie une instance de S3.Bucket créée par la construction.
S3LoggingBucket ?	`s3.Bucket`	Renvoie une instance de S3.Bucket créée par la construction en tant que

Nom	Type	Description
		compartiment de journalisation pour le compartiment principal.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

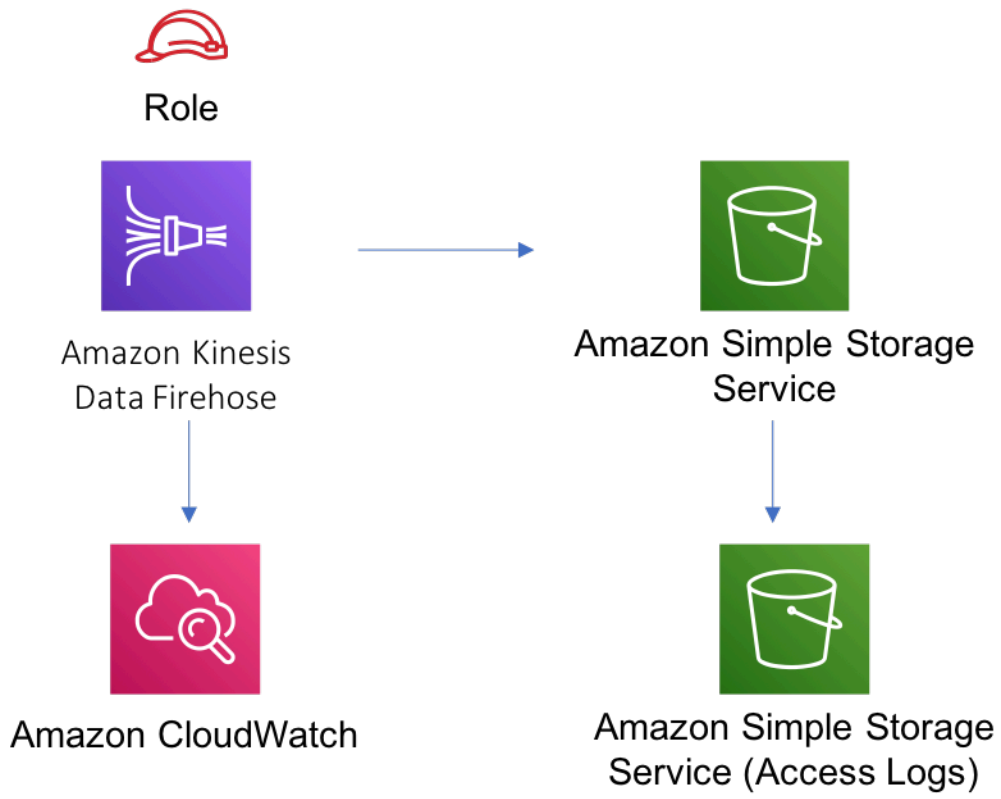
Amazon Kinesis Firehose

- Activer la journalisation CloudWatch pour Kinesis Firehose
- Configurer le rôle IAM d'accès minimal aux privilèges pour Amazon Kinesis Firehose

Bucket Amazon S3

- Configurer la journalisation d'accès pour le compartiment S3
- Activer le chiffrement côté serveur pour le compartiment S3 à l'aide de la clé KMS gérée par AWS
- Activer le contrôle de version pour S3 Bucket
- Ne pas autoriser l'accès public pour le compartiment S3
- Conserver le compartiment S3 lors de la suppression de la pile CloudFormation
- Application du chiffrement des données en transit
- Applique la règle de cycle de vie pour déplacer les versions d'objets non actuelles vers le stockage Glacier après 90 jours

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-construction/aws-kinesisfirehose-s3

aws-kinesisfirehose-s3-et-kinesisanalytics

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion de versions sémantiquesModèle. Cela signifie que même si vous pouvez

les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
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

Cette solution AWS Construct implémente un flux de distribution Amazon Kinesis Firehose connecté à un compartiment Amazon S3 et à une application Amazon Kinesis Analytics.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`KinesisFirehoseToAnalyticsAndS3Props`

Accessoires de construction de modèle

Nom	Type	Description
KinesisFireHoseProps ?	`kinesisFirehose.CfnDeliveryStreamProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour le flux de distribution Kinesis Firehose.
KinesisAnalyticsProps ?	`kinesisAnalytics.CfnApplicationProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de l'application Kinesis Analytics.
Bucketobj existant ?	`s3.IBucket`	Instance existante de l'objet S3 Bucket. Si cela est fourni, alors fournir également `bucketProps` est une erreur. |
BucketProps ?	`s3.BucketProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du compartiment. Ignoré si`existingBucketObj` est fourni.
LogGroupProps ?	`logs.LogGroupProps`	Accessoires fournis par l'utilisateur facultatifs pour remplacer les accessoires par défaut du groupe de journaux CloudWatch Logs.

Propriétés du modèle

Nom	Type	Description
KinesisAnalytics	`kinesisAnalytics.CfnApplication`	Renvoie une instance de l'application Kinesis Analytics créée par le modèle.
KinesisFireHose	`kinesisfirehose.CfnDeliveryStream`	Renvoie une instance du flux de livraison Kinesis Firehose créé par le modèle.
KineSisFireHoselogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle auquel les journaux d'accès Kinesis Firehose sont envoyés.
KineSisFireHoserole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle pour le flux de livraison Kinesis Firehose.
S3Bucket ?	`s3.Bucket`	Renvoie une instance du compartiment S3 créé par le modèle.
S3LoggingBucket ?	`s3.Bucket`	Renvoie une instance du compartiment de journalisation créé par le modèle pour le compartiment S3.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Amazon Kinesis Firehose

- Activer la journalisation CloudWatch pour Kinesis Firehose
- Configurer le rôle IAM d'accès minimal aux privilèges pour Amazon Kinesis Firehose

Amazon S3.

- Configurer la journalisation d'accès pour le compartiment S3
- Activer le chiffrement côté serveur pour le compartiment S3 à l'aide de la clé KMS gérée par AWS
- Activer le contrôle de version pour S3 Bucket
- Ne pas autoriser l'accès public pour le compartiment S3
- Conserver le compartiment S3 lors de la suppression de la pile CloudFormation
- Application du chiffrement des données en transit
- Applique la règle de cycle de vie pour déplacer les versions d'objets non actuelles vers le stockage Glacier après 90 jours

Amazon Kinesis Data Analytics

- Configurer le rôle IAM d'accès minimal aux privilèges pour Amazon Kinesis Analytics

Architecture

GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

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

aws-kinesisstreams-gluejob

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Ceux-ci ne sont pas assujettis à laVersion sémantiqueModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Cette solution AWS Construct déploie un flux de données Amazon Kinesis et configure une Job AWS Glue pour effectuer une transformation ETL personnalisée avec les ressources/propriétés appropriées pour l'interaction et la sécurité. Il crée également un compartiment Amazon S3 où le script Python pour AWS Glue Job peut être téléchargé.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`KinesisstreamsToGluejobProps`

Accessoires de construction de modèle

Nom	Type	Description
KinesisStreamProps ?	`kinesis.StreamProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut du flux de données Amazon Kinesis.
L'existence de Streamobj ?	`kinesis.Stream`	Instance existante de Kinesis Stream, fournissant à la fois ceci et`kinesisStreamProps` provoquera une erreur.
GlueJobProps ?	`cfnJob.CfnJobProps`	Props fournis par l'utilisateur pour remplacer les accessoires par défaut pour le travail AWS Glue.
ExistementGlueJob ?	`cfnJob.CfnJob`	Instance existante d'AWS Glue Job, fournissant à la fois ceci et`glueJobProps` provoquera une erreur.
Une base de données existante ?	`CfnDatabase`	Base de données AWS Glue existante à utiliser avec cette construction. Si cela est défini, alors`databaseProps` est ignoré.
DatabaseProps ?	`CfnDatabaseProps`	Props fournis par l'utilisateur pour remplacer les accessoires par défaut utilisés pour créer la base de données AWS Glue.

Nom	Type	Description
Table existante ?	`CfnTable`	Instance existante de la table AWS Glue. Si cela est défini, alors`tableProps` and`fieldSchema` sont ignorés.
TableProps ?	`CfnTableProps`	Props fournis par l'utilisateur pour remplacer les accessoires par défaut utilisés pour créer une table AWS Glue.
FieldSchema ?	`CfnTable.ColumnProperty[]`	Structure de schéma fournie par l'utilisateur pour créer une table AWS Glue.
Sortie Datastore ?	`SinkDataStoreProps`	Accessoires fournis par l'utilisateur pour un compartiment Amazon S3 qui stocke la sortie de la tâche AWS Glue. Actuellement, Amazon S3 ne prend en charge que le type de banque de données en sortie.

SinkDataStoreProps

Nom	Type	Description
Existants3OutputBucket ?	`Bucket`	Instance existante du compartiment S3 dans laquelle les données doivent être écrites. Fournir à la fois ceci et`outputBuc`

Nom	Type	Description
		`ketProps` provoquera une erreur.
OutputBucketProps	`BucketProps`	Propriétés de compartiment fournies par l'utilisateur pour créer le compartiment Amazon S3 utilisé pour stocker la sortie du travail AWS Glue.
DatasToreType	`SinkStoreType`	Type de magasin de données de lavabo.

SinkStoreType

Énumération des types de stockage de données pouvant inclure S3, DynamoDB, DocumentDB, RDS ou Redshift. L'implémentation de construction actuelle ne prend en charge que S3, mais il est possible d'ajouter d'autres types de sortie à l'avenir.

Nom	Type	Description
S3	`string`	Type de stockage S3

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Flux Amazon Kinesis

- Configurez le rôle IAM d'accès le moins privilégié pour le flux de données Amazon Kinesis.
- Activez le chiffrement côté serveur pour Amazon Kinesis Stream à l'aide d'une clé KMS gérée AWS.
- Déployez les meilleures pratiques Amazon CloudWatch Alarmes pour Amazon Kinesis Stream.

Job de Glue

- Créez une configuration de sécurité AWS Glue qui configure le chiffrement pour CloudWatch, Job Bookmarks et S3. CloudWatch et Job Bookmarks sont chiffrés à l'aide de la clé KMS gérée AWS créée pour AWS Glue Service. Le compartiment S3 est configuré avec le mode de chiffrement SSE-S3.
- Configurez des stratégies de rôle de service qui permettent à AWS Glue de lire depuis Amazon Kinesis Data Streams.

Base de données glue

- Créez une base de données AWS Glue. Une table AWS Glue sera ajoutée à la base de données. Ce tableau définit le schéma des enregistrements mis en mémoire tampon dans le flux de données Amazon Kinesis.

Table de Glue

- Créez une table AWS Glue. La définition du schéma de table est basée sur la structure JSON des enregistrements mis en mémoire tampon dans le flux de données Amazon Kinesis.

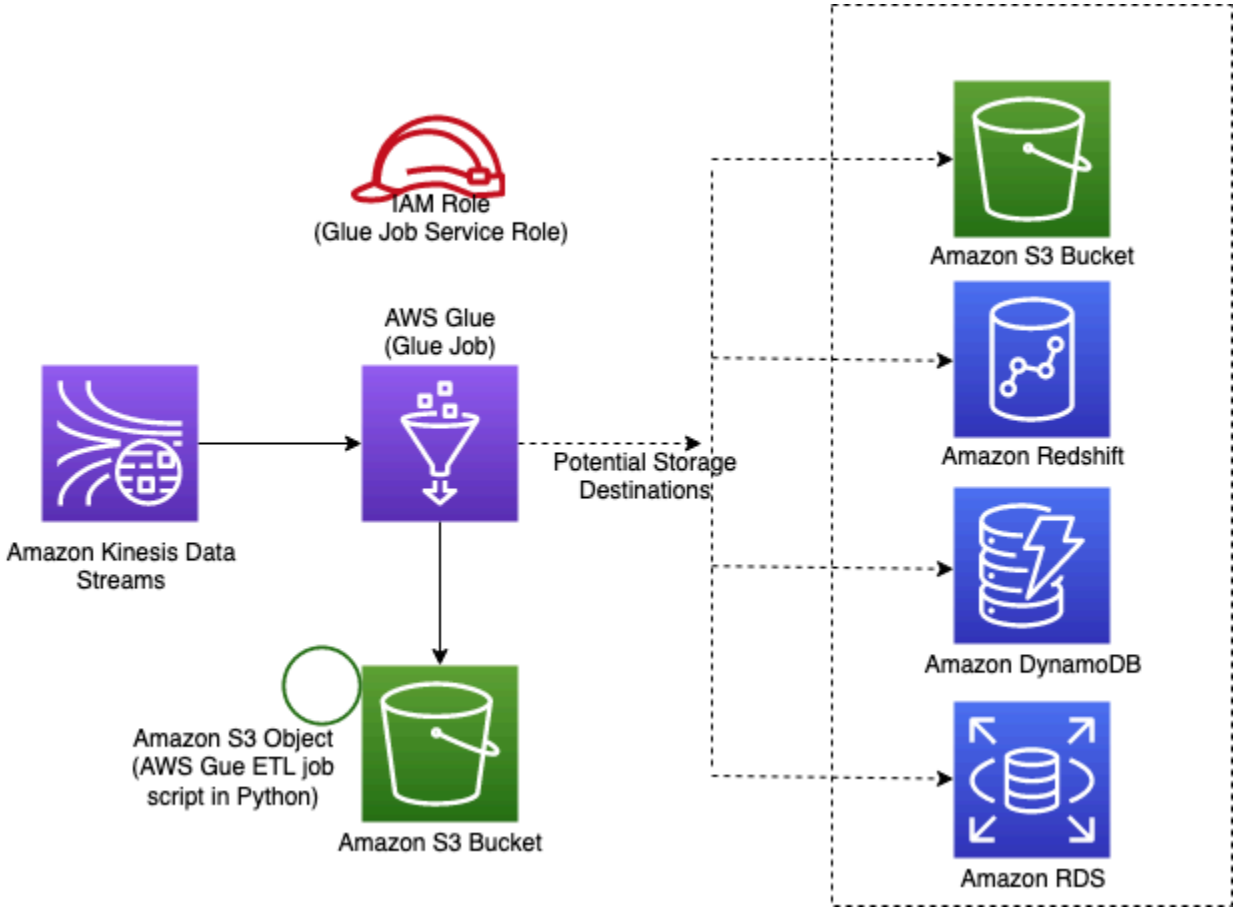
Rôle IAM

- Rôle d'exécution de tâche qui dispose des privilèges suivants : 1) lecture du script ETL à partir de l'emplacement du compartiment Amazon S3, 2) lecture des enregistrements du flux de données Amazon Kinesis et 3) exécution du travail Amazon Glue.

S3 de sortie

- Compartiment Amazon S3 où stocker la sortie de la transformation ETL. Ce compartiment sera transmis en tant qu'argument au travail AWS Glue créé afin qu'il puisse être utilisé dans le script ETL pour y écrire des données.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-construction/aws-kinesiss treams-gluejob

aws-kinesisstreams-kinesisfirehose-s3

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Ceux-ci ne sont pas assujettis à laGestion sémantique des versionsModèle. Cela signifie que même si vous pouvez les

utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_kinesisstreams_kinesisfirehose_s3`
TypeScript	`@aws-solutions-constructs/aws-kinesis-streams-kinesis-firehose-s3`
Java	`software.amazon.awsconstructs.services.kinesisstreamskinesisfirehoses3`

Overview

Cette solution AWS Construct implémente un flux de données Amazon Kinesis (KDS) connecté au flux de distribution Amazon Kinesis Data Firehose (KDF) connecté à un compartiment Amazon S3.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

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

Accessoires de construction de modèle

Nom	Type	Description
BucketProps ?	`s3.BucketProps`	Des accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut pour le compartiment S3.
Créer CloudWatchArms ?	`boolean`	Facultatif si vous souhaitez créer des alarmes CloudWatch recommandées.
Bucketobj existant ?	`s3.IBucket`	Instance existante facultative de l'objet S3 Bucket. Si cela est fourni, alors fournir également`bucketProps` est une erreur.
ExistingLoggingBucketObj ?	`s3.IBucket`	Instance existante facultative de journalisation de l'objet S3 Bucket pour le compartiment S3 créé par le modèle.
L'existence de Streamobj ?	`kinesis.Stream`	Instance existante de Kinesis Stream, fournissant à la fois ceci et`kinesisSt`

Nom	Type	Description
		`reamProps` provoquera une erreur.
KinesisFireHoseProps ?	`aws-kinesisfirehose.CfnDeliveryStreamProps` | any	Des accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut pour Kinesis Firehose Delivery Stream.
KinesisStreamProps ?	`kinesis.StreamProps`	L'utilisateur a fourni des accessoires facultatifs pour remplacer les accessoires par défaut pour le flux Kinesis.
LogGroupProps ?	`logs.LogGroupProps`	Des accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut pour le groupe de journaux CloudWatchLogs.

Propriétés du modèle

Nom	Type	Description
CloudwatAlarm ?	`cloudwatch.Alarm[]`	Renvoie une liste des instances CloudWatch.Alarm créées par la construction.
KinesisFireHose	`kinesisfirehose.CfnDeliveryStream`	Renvoie une instance de KinesisFireHose.cfnDelivery Stream créée par la construct ion.
KineSisFireHoselogGroup	`logs.LogGroup`	Renvoie une instance du logs.logGroup créé par la

Nom	Type	Description
		construction pour le flux de diffusion Kinesis Data Firehose.
KineSisFireHoserole	`iam.Role`	Renvoie une instance de l'IAM.Role créée par la construction pour le flux de livraison Kinesis Data Firehose.
KinesisStreamRole	`iam.Role`	Renvoie une instance de l'IAM.Role créée par la construction pour le flux Kinesis.
S3 Bucket ?	`s3.Bucket`	Renvoie une instance de S3.Bucket créée par la construction.
S3LoggingBucket ?	`s3.Bucket`	Renvoie une instance de S3.Bucket créée par la construction en tant que compartiment de journalisation pour le compartiment principal.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Amazon Kinesis Stream

- Configurer le rôle IAM d'accès minimal aux privilèges pour Kinesis Stream
- Activer le chiffrement côté serveur pour Kinesis Stream à l'aide de la clé KMS gérée AWS
- Déployer les meilleures pratiques des alarmes CloudWatch pour Kinesis Stream

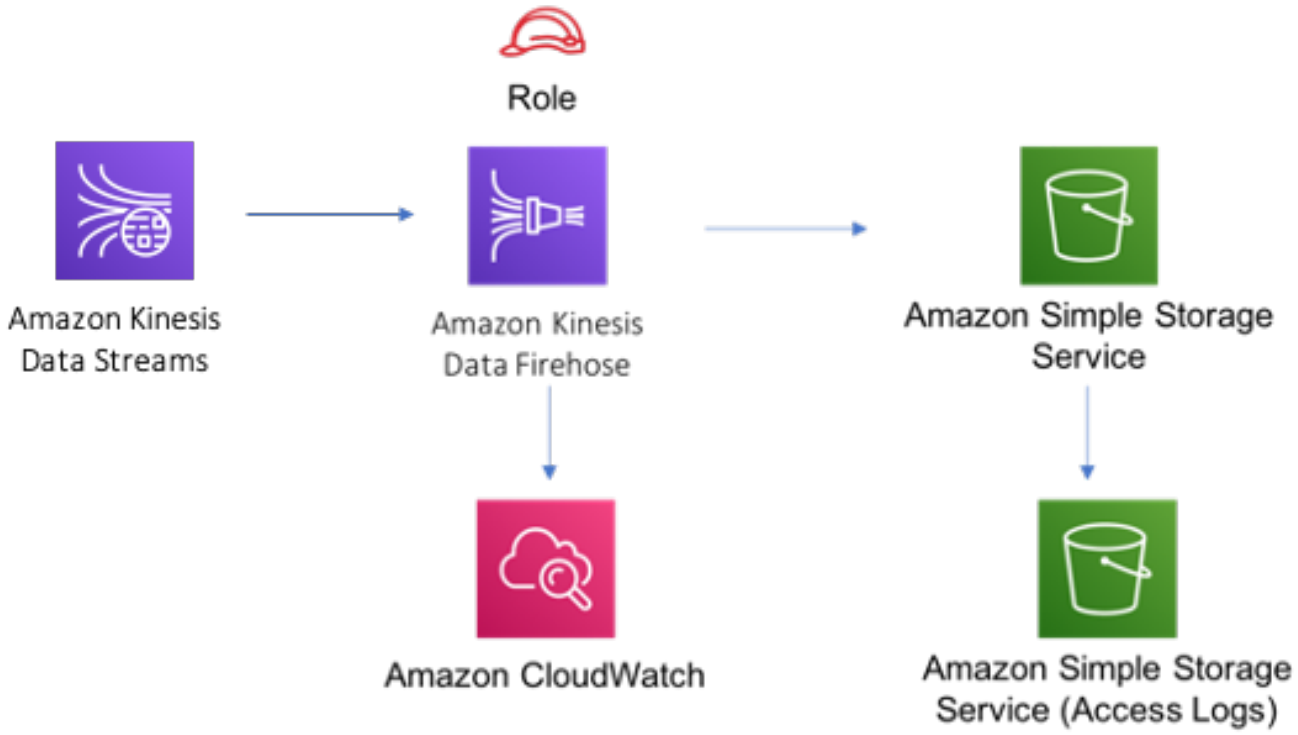
Amazon Kinesis Firehose

- Activer la journalisation CloudWatch pour Kinesis Firehose
- Configurer le rôle IAM d'accès minimal aux privilèges pour Amazon Kinesis Firehose

Amazon S3 Bucket

- Configuration de la journalisation des accès pour le compartiment S3
- Activer le chiffrement côté serveur pour le compartiment S3 à l'aide de la clé KMS gérée par AWS
- Application du chiffrement des données en transit
- Activer le contrôle de version du compartiment
- Ne pas autoriser l'accès public pour le compartiment S3
- Conserver le compartiment S3 lors de la suppression de la pile CloudFormation
- Appliquer une règle de cycle de vie pour déplacer des versions d'objets non actuelles vers le stockage Glacier après 90 jours

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-construction/aws-kinesisstreams-kinesisfirehose-s3

aws-kinesisstreams-lambda

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Ceux-ci ne sont pas assujettis à laGestion sémantique de versionModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

AWS Solutions Construct déploie une fonction Kinesis Stream et Lambda avec les ressources/propriétés appropriées pour l'interaction et la sécurité.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`KinesisStreamsToLambdaProps`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.
KinesisStreamProps ?	`kinesis.StreamProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour le flux Kinesis.
L'existence de Streamobj ?	`kinesis.Stream`	Instance existante de Kinesis Stream, fournissant à la fois ceci et`kinesisStreamProps` provoquera une erreur.
KinesisEventSourceProps ?	`aws-lambda-event-sources.KinesisEventSourceProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour le mappage de source d'événement Lambda.
CreateCloudWatchArms	`boolean`	Indique s'il faut créer des alarmes CloudWatch recommandées.

Propriétés de modèle

Nom	Type	Description
Stream KinesisStream	`kinesis.Stream`	Renvoie une instance du flux Kinesis créé par le modèle.
LambdaFonction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.
KinesisStreamRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle pour le flux Kinesis.
Cloudwatch Alarm ?	`cloudwatch.Alarm[]`	Renvoie la liste d'une ou plusieurs alarmes CloudWatch créées par le modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Amazon Kinesis Stream

- Configurez le rôle IAM d'accès le moins élevé pour Kinesis Stream.
- Activez le chiffrement côté serveur pour Kinesis Stream à l'aide de la clé KMS gérée par AWS.
- Déployez les meilleures pratiques des alarmes CloudWatch pour Kinesis Stream.

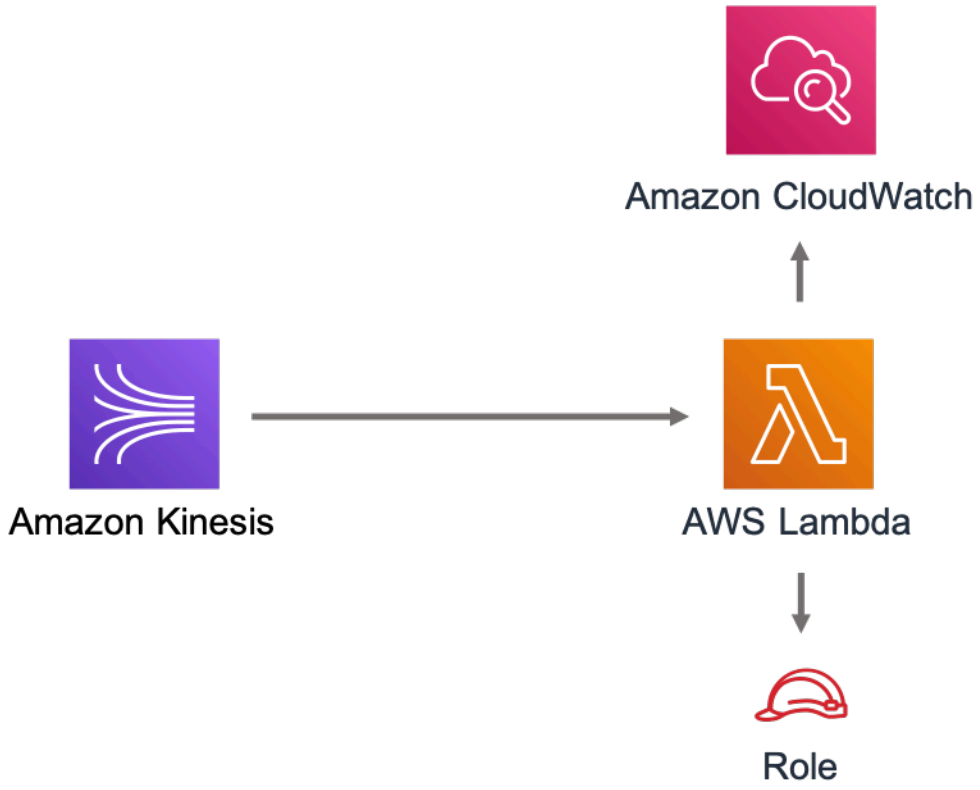
Fonction AWS Lambda

- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Activer le suivi X-Ray.
- Activer les fonctionnalités de gestion des défaillances : activer le bisect sur la fonction Erreur ; définir l'âge maximal des enregistrements par défaut (24 heures) ; définir les tentatives de relance

maximales par défaut (500) ; et déployer la file d'attente des lettres mortes SQS comme destination en cas d'échec.

- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-construction/aws-kinesisstreams-lambda

aws-lambda-dynamodb

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Ceux-ci ne sont pas assujettis à laGestion sémantique de versionsmodèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_lambda_dynamodb`
Typecript	`@aws-solutions-constructs/aws-lambda-dynamodb`
Java	`software.amazon.awsconstructs.services.lambdadynamodb`

Overview

Cette solution AWS Construct implémente la fonction AWS Lambda et la table Amazon DynamoDB avec les autorisations les moins privilèges.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`LambdaToDynamoDBProps`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.
DynamoTableProps ?	`dynamodb.TableProps`	Props fournis par l'utilisateur en option pour remplacer les

Nom	Type	Description
		accessoires par défaut pour DynamoDB Table
ExistantTableObj ?	`dynamodb.Table`	Instance existante de l'objet de table DynamoDB, fournissant à la fois ceci et`dynamoTableProps` provoquera une erreur.
TableAutorisations ?	`string`	Autorisations de table facultatives à accorder à la fonction Lambda. L'une des options suivantes peut être spécifiée :`All`,`Read`,`ReadWrite`, ou`Write`.
TableEnvironmentVariableName ?	`string`	Nom facultatif de la variable d'environnement de table DynamoDB définie pour la fonction Lambda.

Nom	Type	Description
VPC existant ?	`ec2.IVpc`	Un VPC existant optionnel dans lequel ce modèle doit être déployé. Lorsqu'elle est déployée dans un VPC, la fonction Lambda utilise ENI dans le VPC pour accéder aux ressources réseau et un point de terminaison de passerell e est créé dans le VPC pour Amazon DynamoDB. Si un VPC existant est fourni, le`deployVpc` ne peut pas être`true`. Cela utilise`ec2.IVpc`pour permettre aux clients de fournir des VPC qui existent en dehors de la pile à l'aide de la méthode`ec2.Vpc.fromLookup()` Méthode.
VPCProps ?	`ec2.VpcProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du nouveau VPC.`enableDnsHostnames` ,`enableDnsSupport` ,`natGateways` , et`subnetConfiguration` sont définies par le modèle, donc toutes les valeurs pour ces propriétés fournies ici seront remplacées. Si`deployVpc` n'est pas`true`Cette propriété sera ignorée.

Nom	Type	Description
Déploiement de VPC ?	`boolean`	Que ce soit pour créer un VPC basé sur`vpcProps`dans lequel déployer ce modèle. La définition de cette valeur sur true déploiera le VPC minimal et le plus privé pour exécuter le modèle : • Un sous-réseau isolé dans chaque zone de disponibilité utilisée par le programme CDK • `enableDnsHostnames` and`enableDns Support` seront tous les deux réglés sur`true` Si cette propriété est`true`, puis`existingVpc` ne peut pas être spécifié. La valeur par défaut est `false`.

Propriétés du modèle

Nom	Type	Description
DynamoTable	`dynamodb.Table`	Renvoie une instance de la table DynamoDB créée par le modèle.
LambdaUNction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.

Nom	Type	Description
VPC ?	`ec2.IVpc`	Renvoie une interface sur le VPC utilisé par le modèle (le cas échéant). Il peut s'agir d'un VPC créé par le modèle ou du VPC fourni au construct eur de modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

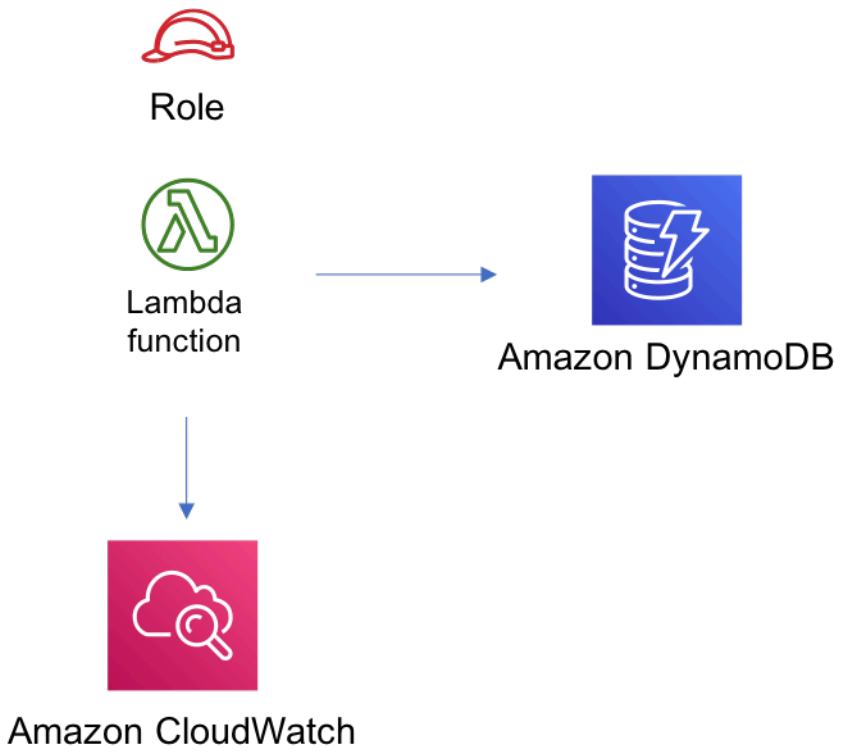
Fonction AWS Lambda

- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Activer le suivi X-Ray.
- Définir les variables d'environnement :
  - `DDB_TABLE_NAME` (default)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Amazon DynamoDB Table

- Définissez le mode de facturation de la table DynamoDB sur On-Demand (Paiement par demande).
- Activez le chiffrement côté serveur pour la table DynamoDB à l'aide de la clé KMS gérée par AWS.
- Crée une clé de partition appelée 'id' pour DynamoDB Table.
- Conservez la table lors de la suppression de la pile CloudFormation.
- Activez les sauvegardes continues et la restauration à un instant dans le passé.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-constructions/aws-lambda-dynamodb

aws-lambda-elasticsearch-kibana

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion sémantique de versionModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

AWS Solutions Construct implémente une fonction AWS Lambda et un domaine Amazon Elasticsearch Service avec les autorisations les moins privilégiées.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`LambdaToElasticSearchAndKibanaProps`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.
ESDomainProps ?	`elasticsearch.CfnDomainProps`	Props fournis par l'utilisateur en option pour remplacer les

Nom	Type	Description
		accessoires par défaut pour Amazon Elasticsearch Service
domainName	`string`	Nom de domaine pour Cognito et Amazon Elasticsearch Service
CognitoNom de domaine ?	`string`	Nom de domaine Cognito facultatif. S'il est fourni, il sera utilisé pour le domaine Cognito, et`domainName` sera utilisé pour le domaine Elasticsearch.
CreateCloudWatchArms	`boolean`	Indique s'il faut créer des alarmes CloudWatch recommandées.
DomainEndPointEnVironmentVariableName ?	`string`	Nom facultatif pour la variable d'environnement de point de terminaison de domaine ElasticSearch définie pour la fonction Lambda.

Propriétés du modèle

Nom	Type	Description
Cloudwatch Alarm ?	`cloudwatch.Alarm[]`	Renvoie la liste d'une ou plusieurs alarmes CloudWatch créées par le modèle.
ElasticSearchDomain	`elasticsearch.CfnDomain`	Renvoie une instance du domaine Elasticsearch créé par le modèle.

Nom	Type	Description
ElasticSearchDomainRole	`iam.Role`	Renvoie une instance du rôle IAM créé par le modèle pour le domaine Elasticsearch.
IdentityPool	`cognito.CfnIdentityPool`	Renvoie une instance du pool d'identités Cognito créé par le modèle.
LambdaFunction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.
userPool	`cognito.UserPool`	Renvoie une instance du pool d'utilisateurs Cognito créé par le modèle.
UserPoolClient	`cognito.UserPoolClient`	Renvoie une instance du client de pool d'utilisateurs Cognito créé par le modèle.

Fonction Lambda

Ce modèle nécessite une fonction Lambda qui peut publier des données dans le service Elasticsearch à partir du flux DynamoDB. Un exemple de fonction est fourniIci.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Fonction AWS Lambda

- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Activer le suivi X-Ray.
- Définir les variables d'environnement :

- `DOMAIN_ENDPOINT` (default)
- `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

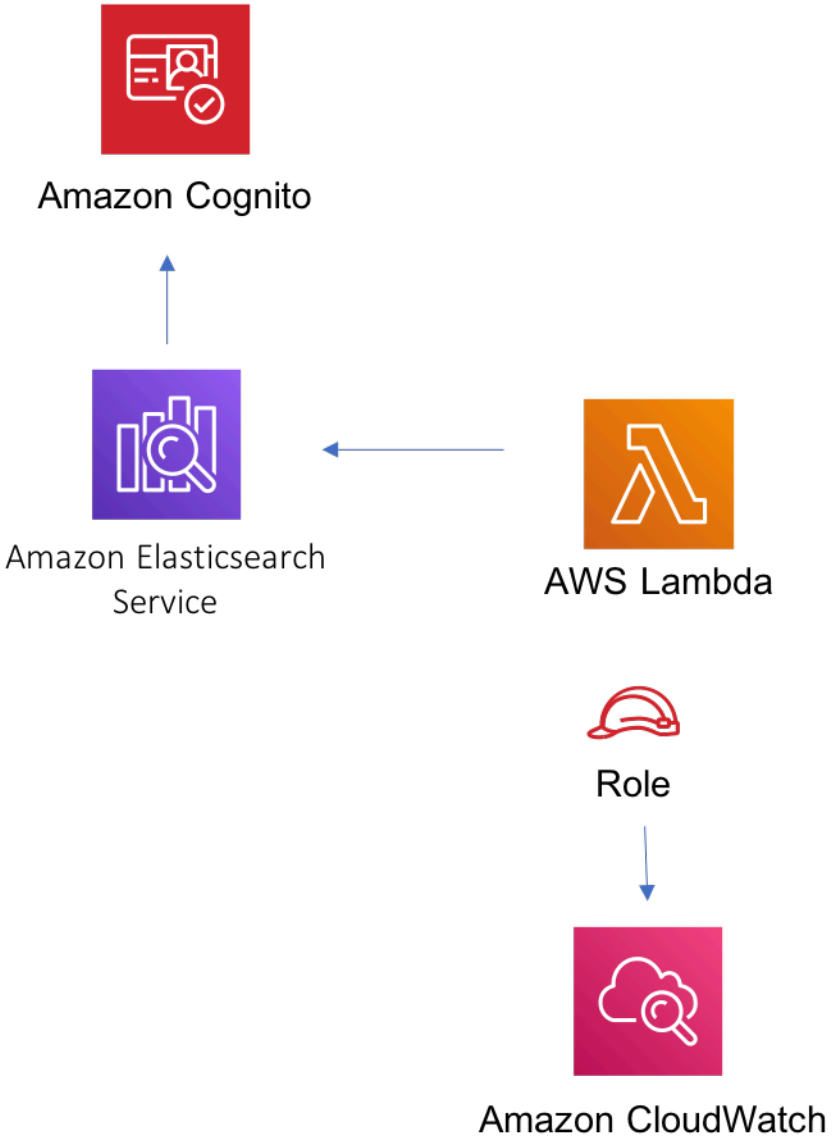
Amazon Cognito

- Définir la stratégie de mot de passe pour les pools d'utilisateurs
- Appliquez le mode de sécurité avancé pour les pools d'utilisateurs.

Amazon Elasticsearch Service

- Déployez les meilleures pratiques CloudWatch Alarmes pour le domaine Elasticsearch.
- Sécurisez l'accès au tableau de bord Kibana avec les pools d'utilisateurs Cognito.
- Activez le chiffrement côté serveur pour le domaine Elasticsearch à l'aide de la clé KMS gérée par AWS.
- Activer le chiffrement nœud à nœud pour le domaine Elasticsearch.
- Configurez le cluster pour le domaine Amazon ES.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructs/aws-lambda-elasticsearch-kibana

aws-lambda-s3

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion sémantique de versionModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Ce kit AWS Solutions Construct implémente une fonction AWS Lambda connectée à un compartiment Amazon S3.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`LambdaToS3Props`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.
Bucketobj existant ?	`s3.IBucket`	Instance existante de l'objet S3 Bucket. Si cela est fourni,

Nom	Type	Description
		alors fournir également `bucketProps` est une erreur.
BucketProps ?	`s3.BucketProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du compartiment. Ignoré si `unexistingBucketObj` est fourni.
Autorisations BucketPermissions ?	`string[]`	Autorisations de compartiment facultatives à accorder à la fonction Lambda. Un ou plusieurs des éléments suivants peuvent être spécifiés :`Delete`,`Put`,`Read`,`ReadWri` ,`Write`.

Nom	Type	Description
VPC existant ?	`ec2.IVpc`	Un VPC existant optionnel dans lequel ce modèle doit être déployé. Lorsqu'elle est déployée dans un VPC, la fonction Lambda utilise ENI dans le VPC pour accéder aux ressources réseau et un point de terminais on d'interface est créé dans le VPC pour Amazon SQS. Si un VPC existant est fourni, le`deployVpc` ne peut pas être`true`. Cela utilise`ec2.IVpc`pour permettre aux clients de fournir des VPC qui existent en dehors de la pile à l'aide de la méthode`ec2.Vpc.fromLookup()` Méthode.

Nom	Type	Description
Déploiement de VPC ?	`boolean`	S'il faut créer un VPC basé sur`vpcProps`dans lequel déployer ce modèle. Définisse z sur`true`déploiera le VPC minimal et le plus privé pour exécuter le modèle : • Un sous-réseau isolé dans chaque zone de disponibilité utilisée par le programme CDK. • `enableDnsHostnames` and`enableDnsSupport` seront tous les deux définis sur`true`. Si cette propriété est`true`, puis`existingVpc` ne peut pas être spécifié. La valeur par défaut est `false`.

Nom	Type	Description
VPCProps ?	`ec2.VpcProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du nouveau VPC.`enableDnsHostnames` ,`enableDnsSupport` ,`natGateways` and`subnetConfiguration` sont définies par le modèle, donc toutes les valeurs pour ces propriétés fournies ici seront remplacées. Si`deployVpc` n'est pas`true`Cette propriété sera ignorée.
BucketEnvironmentVariableName ?	`string`	Nom facultatif de la variable d'environnement de compartiment S3 définie pour la fonction Lambda.

Propriétés du modèle

Nom	Type	Description
LambdaFunction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.
S3Bucket ?	`s3.Bucket`	Renvoie une instance du compartiment S3 créé par le modèle.
S3LoggingBucket ?	`s3.Bucket`	Renvoie une instance du compartiment de journalisation

Nom	Type	Description
		créé par le modèle pour le compartiment S3.
Un vpc ?	`ec2.IVpc`	Renvoie une instance du VPC utilisé par le modèle (le cas échéant). Il peut s'agir d'un VPC créé par le modèle ou du VPC fourni au constructeur de modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Fonction AWS Lambda

- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Activer le suivi X-Ray
- Définissez les variables d'environnement :
  - `S3_BUCKET_NAME` (default)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Bucket Amazon S3

- Configurez la journalisation d'accès pour le compartiment S3.
- Activez le chiffrement côté serveur pour le compartiment S3 à l'aide de la clé KMS gérée par AWS.
- Activez le contrôle de version pour S3 Bucket.
- N'autorisez pas l'accès public pour le compartiment S3.
- Conservez le compartiment S3 lors de la suppression de la pile CloudFormation.
- Application du chiffrement des données en transit.
- Applique la règle de cycle de vie pour déplacer les versions d'objets non actuelles vers le stockage Glacier après 90 jours.

Architecture

GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-construction/aws-lambda-s3

aws-lambda-ssmstringparameter

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion de version sémantiqueModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Ce module AWS Solutions Construct implémente la fonction AWS Lambda et le paramètre Chaîne de stockage de paramètres AWS Systems Manager avec les autorisations les moins privilégiées.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`LambdaToSsmstringparameterProps`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.
ExistingStringParameterObJ ?	`ssm.StringParameter`	Instance existante de l'objet de paramètre SSM String, fournissant à la fois ceci et`stringParameterProps` provoquera une erreur.
StringParameterProps ?	`ssm.StringParameterProps`	Des accessoires facultatifs fournis par l'utilisateur pour

Nom	Type	Description
		remplacer les accessoires par défaut pour le paramètre SSM String. Si `existingStringParameterObj` n'est pas défini, `stringParameterProps` Le paramètre est obligatoire. Le seul pris en charge `ssm.StringParameterProps.type` est `STRING` si une valeur différente est fournie, elle sera remplacée.
StringParameterEnvironmentVariableName ?	`string`	Nom facultatif de la variable d'environnement de paramètre SSM String définie pour la fonction Lambda.

Nom	Type	Description
VPC existant ?	`ec2.IVpc`	Un VPC existant optionnel dans lequel ce modèle doit être déployé. Lorsqu'elle est déployée dans un VPC, la fonction Lambda utilise les ENI du VPC pour accéder aux ressources réseau et un point de terminaison d'interfa ce est créé dans le paramètre VPC for AWS Systems Manager. Si un VPC existant est fourni, le`deployVpc` ne peut pas être`true`. Cela utilise`ec2.IVpc`pour permettre aux clients de fournir des VPC qui existent en dehors de la pile à l'aide de la méthode`ec2.Vpc.fromLookup()` Méthode.
VPCProps ?	`ec2.VpcProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du nouveau VPC.`enableDnsHostnames` ,`enableDnsSupport` ,`natGateways` and`subnetConfiguration` sont définies par le modèle, donc toutes les valeurs pour ces propriétés fournies ici seront remplacées s. Si`deployVpc` n'est pas`true`Cette propriété sera ignorée.

Nom	Type	Description
Déploiement de VPC ?	`boolean`	Que ce soit pour créer un VPC basé sur`vpcProps`dans lequel déployer ce modèle. Paramètre sur`true`déploiera le VPC minimal et le plus privé pour exécuter le modèle : • Un sous-réseau isolé dans chaque zone de disponibilité utilisée par le programme CDK. • `enableDnsHostnames` and`enableDns Support` seront tous les deux réglés sur`true`. Si cette propriété est définie sur`true`, puis`existingVpc` ne peut pas être spécifié. La valeur par défaut est `false`.
StringParameterPermissions ?	`string`	Autorisations de paramètre SSM String facultatives à accorder à la fonction Lambda. L'un des éléments suivants peut être spécifié :`Read`,`ReadWrite` .

Propriétés de modèle

Nom	Type	Description
LambdaFonction	`lambda.Function`	Retourne une instance de`lambda.Function` créé par la construction.
StringParameter	`ssm.StringParameter`	Retourne une instance de`ssm.StringParameter` créé par la construction.
VPC ?	`ec2.IVpc`	Renvoie une interface sur le VPC utilisé par le modèle (le cas échéant). Il peut s'agir d'un VPC créé par le modèle ou du VPC fourni au constructeur de modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Fonction AWS Lambda

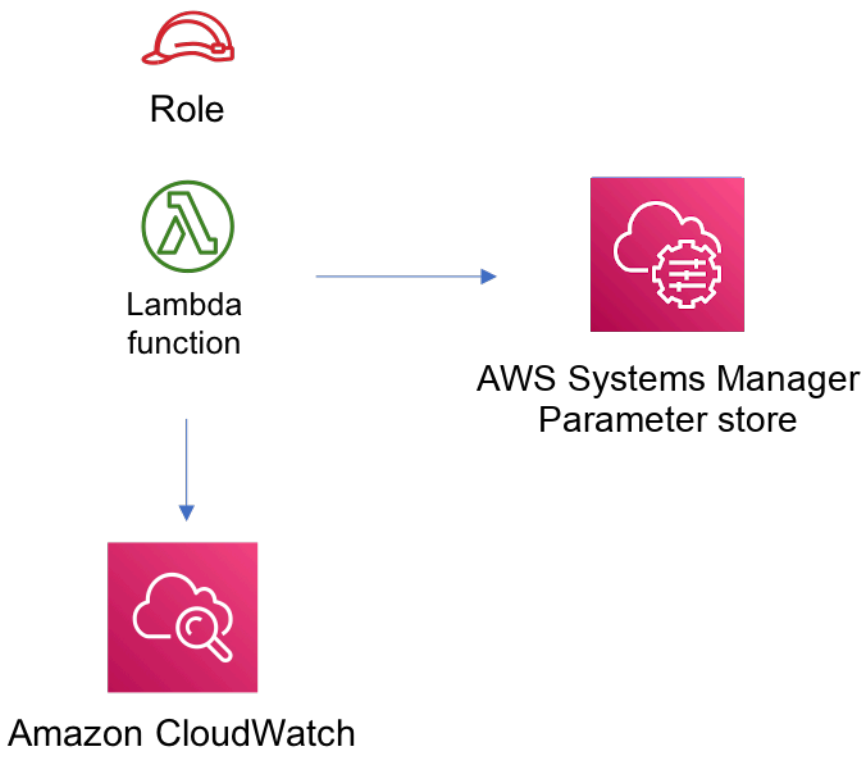
- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Activez le suivi X-Ray.
- Définir les variables d'environnement :
  - `SSM_STRING_PARAMETER_NAME` (default)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Chaîne de stockage de paramètres Amazon AWS Systems Manager

- Activez l'accès en lecture seule pour la fonction AWS Lambda associée.

- Crée un nouveau paramètre SSM String avec les valeurs fournies.
- Conservez le paramètre SSM String lors de la suppression de la pile CloudFormation.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructs/aws-lambda-ssm stringparameter

aws-lambda-sagemakerendpoint

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à

laGestion de versions sémantiquesModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Builts et AWS CDK de votre projet doivent être la même version.

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

Overview

Cette solution AWS Construct implémente une fonction AWS Lambda connectée à un point de terminaison Amazon Sagemaker.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`LambdaToSagemakerEndpointProps`

Modèle de construction

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.

Nom	Type	Description
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda.
ExistantSageMakerendPointTobj ?	`sagemaker.CfnEndpoint`	Un enpoint Sagemaker existant en option à utiliser. Fournir à la fois ceci et`endpointProps` provoquera une erreur.
ModelProps ?	`sagemaker.CfnModelProps` | any	Propriétés fournies par l'utilisateur pour remplacer les propriétés par défaut du modèle Sagemaker . Au moins`modelProps.primaryContainer` doit être fourni pour créer un modèle. Par défaut, le modèle créera un rôle avec les autorisations minimales requises, mais le client peut fournir un rôle personnalisé avec des fonctionnalités supplémentaires en utilisant `modelProps.executionRoleArn` .
EndPointConfigProps ?	`sagemaker.CfnEndpointConfigProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la configuration Sagemaker Endpoint.

Nom	Type	Description
EndPointProps ?	`sagemaker.CfnEndpointProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du point de terminaison Sagemaker.
VPC existant ?	`ec2.IVpc`	Un VPC existant optionnel dans lequel cette construction doit être déployée. Lorsqu'il s sont déployés dans un VPC, la fonction Lambda et Sagemaker Endpoint utilisent les ENI du VPC pour accéder aux ressources réseau. Un point de terminaison d'interface sera créé dans le VPC pour Amazon Sagemaker Runtime et Amazon S3 VPC Endpoint. Si un VPC existant est fourni, le `deployVpc` ne peut pas être `true`.

Nom	Type	Description
VPCProps ?	`ec2.VpcProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du nouveau VPC.`enableDnsHostnames` ,`enableDnsSupport` ,`natGateways` and`subnetConfiguration` sont définis par la construction, donc toutes les valeurs pour ces propriétés fournies ici seront remplacées. Si`deployVpc` n'est pas`true`, cette propriété sera ignorée.

Nom	Type	Description
Déploiement de VPC ?	`boolean`	Que ce soit pour créer un nouveau VPC basé sur`vpcProps`dans lequel déployer ce modèle. Paramètre sur`true`déploiera le VPC minimal et le plus privé pour exécuter le modèle : • Un sous-réseau isolé dans chaque zone de disponibilité utilisée par le programme CDK. • `enableDnsHostnames` and`enableDnsSupport` seront tous les deux définis sur`true`. Si cette propriété est définie sur`true`, puis`existingVpc` ne peut pas être spécifié. La valeur par défaut est `false`.
SageMakerEnvironmentVariableName ?	`string`	Nom facultatif de la variable d'environnement de point de terminaison SageMaker définie pour la fonction Lambda.

Propriétés de modèle

Nom	Type	Description
LambdaUNction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.
SageMakerEndPoint	`sagemaker.CfnEndpoint`	Renvoie une instance du point de terminaison Sagemaker créé par le modèle.
SageMakerEndPointConfig ?	`sagemaker.CfnEndpointConfig`	Renvoie une instance de SageMaker EndpointConfig créée par le modèle, si `existingSagemakerEndpointObj` n'est pas fourni.
SageMakerModel ?	`sagemaker.CfnModel`	Renvoie une instance du modèle Sagemaker créé par le modèle, si `existingSagemakerEndpointObj` n'est pas fourni.
VPC ?	`ec2.IVpc`	Renvoie une instance du VPC créée par le modèle, si `deployVpc` est `true`, ou si `existingVpc` est fourni.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Fonction AWS Lambda

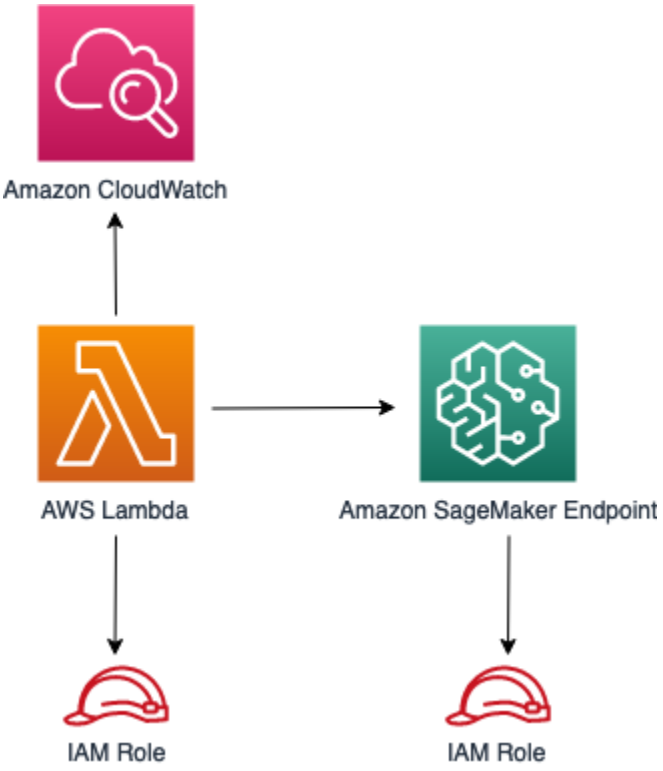
- Configurez le rôle IAM d'accès limité pour la fonction Lambda.

- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Autorisez la fonction à appeler le point de terminaison Sagemaker pour les inférences.
- Configurez la fonction pour accéder aux ressources du VPC, où le point de terminaison Sagemaker est déployé.
- Activez le suivi X-Ray
- Définir les variables d'environnement :
  - `SAGEMAKER_ENDPOINT_NAME` (default)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Amazon SageMaker

- Configurez des privilèges limités pour créer des ressources Sagemaker.
- Déployez le modèle Sagemaker, EndPointConfig et le point de terminaison.
- Configurez le point de terminaison Sagemaker à déployer dans un VPC.
- Déployez le point de terminaison VPC S3 et l'interface VPC Runtime de Sagemaker.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-construction/aws-lambda-sagemakerendpoint

aws-lambda-secretsmanager

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion de versions sémantiquesLe modèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Cette solution AWS Construct implémente la fonction AWS Lambda et le secret AWS Secrets Manager avec les autorisations les moins privilégiées.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`LambdaToSecretsmanagerProps`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	L'utilisateur a fourni des accessoires pour remplacer les accessoires par défaut pour la fonction Lambda.
SecretProps ?	`secretsmanager.SecretProps`	Les accessoires facultatifs fournis par l'utilisateur pour remplacer les accessoires par défaut pour Secrets Manager.
SecretoBJ existe-t-il ?	`secretsmanager.Secret`	Instance existante de l'objet secret Secrets Manager, Si cela est défini, alors la propriété`secretProps` est ignoré.
GrantwriteAccess ?	`boolean`	Accès facultatif en écriture au secret pour la fonction Lambda (lecture seule par défaut).
SecretEnVironmentNomVariableName ?	`string`	Nom facultatif de la variable d'environnement secrète Secrets Manager définie pour la fonction Lambda.
VPC existant ?	`ec2.IVpc`	Un VPC existant optionnel dans lequel ce modèle doit

Nom	Type	Description
		être déployé. Lorsqu'elle est déployée dans un VPC, la fonction Lambda utilise les ENI du VPC pour accéder aux ressources réseau et un point de terminaison d'interface est créé dans le VPC for AWS Secrets Manager. Si un VPC existant est fourni, le`deployVpc` ne peut pas être`true`. Cela utilise`ec2.IVpc`pour permettre aux clients de fournir des VPC qui existent en dehors de la pile à l'aide de la méthode`ec2.Vpc.fromLookup()` Méthode.
VPCProps ?	`ec2.VpcProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du nouveau VPC.`enableDnsHostnames` ,`enableDnsSupport` ,`natGateways` , et`subnetConfiguration` sont définies par le modèle, donc toutes les valeurs pour ces propriétés fournies ici seront remplacées. Si`deployVpc` n'est pas`true`Cette propriété sera ignorée.

Nom	Type	Description
Déploiement de VPC ?	`boolean`	Que ce soit pour créer un VPC basé sur`vpcProps`dans lequel déployer ce modèle. Paramètre sur`true`déploiera le VPC minimal et le plus privé pour exécuter le modèle : • Un sous-réseau isolé dans chaque zone de disponibilité utilisée par le programme CDK • `enableDnsHostnames` `andenableDns` `Support` seront tous les deux réglés sur`true` Si cette propriété est`true`, puis`existingVpc` ne peut pas être spécifié. La valeur par défaut est `false`.

Propriétés du modèle

Nom	Type	Description
LambdaFonction	`lambda.Function`	Retourne une instance de`lambda.Function` créé par la construction.
secret	`secretsmanager.Secret`	Retourne une instance de`secretsmanager.Secret` créé par la construction.

Nom	Type	Description
VPC ?	`ec2.IVpc`	Renvoie une interface sur le VPC utilisé par le modèle (le cas échéant). Il peut s'agir d'un VPC créé par le modèle ou du VPC fourni au construct eur de modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

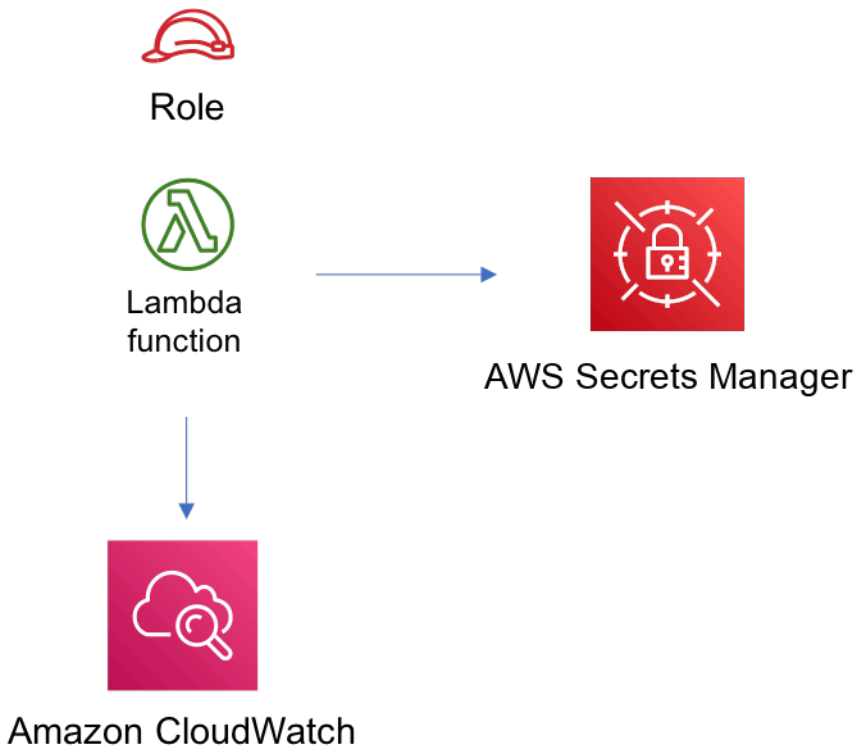
Fonction AWS Lambda

- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Activez le suivi X-Ray.
- Définissez les variables d'environnement :
  - (par défaut) SECRET_ARN contenant l'ARN du secret comme retour par CDKSecretArnLa propriété
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Secret Amazon Secrets Manager

- Activer l'accès en lecture seule pour la fonction AWS Lambda associée
- Activer le chiffrement côté serveur à l'aide d'une clé KMS par défaut pour le compte et la région
- Nous créons un secret :
  - (par défaut) nom aléatoire
  - valeur aléatoire (par défaut)
- Conserver le secret lors de la suppression de la pile CloudFormation

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructs/aws-lambda-secretsmanager

aws-lambda-sns

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion de versions sémantiquesLe modèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Cette AWS Solutions Construct implémente une fonction AWS Lambda connectée à une rubrique Amazon SNS.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`LambdaToSnsProps`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.
ExistantTopicObj ?	`sns.Topic`	Instance existante de l'objet SNS Topic, fournissant à la fois ceci et`topicProps` provoquera une erreur.
SujetProps ?	`sns.TopicProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer

Nom	Type	Description
		les propriétés par défaut de la rubrique SNS.
VPC existant ?	`ec2.IVpc`	Un VPC existant optionnel dans lequel ce modèle doit être déployé. Lorsqu'elle est déployée dans un VPC, la fonction Lambda utilise ENI dans le VPC pour accéder aux ressources réseau et un point de terminais on d'interface est créé dans le VPC pour Amazon SQS. Si un VPC existant est fourni, le`deployVpc` ne peut pas être`true`. Cela utilise`ec2.IVpc`pour permettre aux clients de fournir des VPC qui existent en dehors de la pile à l'aide de la méthode`ec2.Vpc.fromLookup()` Méthode.

Nom	Type	Description
Déploiement de VPC ?	`boolean`	Que ce soit pour créer un VPC basé sur`vpcProps`dans lequel déployer ce modèle. Paramètre sur`true`déploiera le VPC minimal et le plus privé pour exécuter le modèle : • Un sous-réseau isolé dans chaque zone de disponibilité utilisée par le programme CDK. • `enableDnsHostnames` and`enableDns Support` seront tous les deux réglés sur`true`. Si cette propriété est`true`, puis`existingVpc` ne peut pas être spécifié. La valeur par défaut est `false`.

Nom	Type	Description
VPCProps ?	`ec2.VpcProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du nouveau VPC.`enableDnsHostnames` ,`enableDnsSupport` ,`natGateways` and`subnetConfiguration` sont définies par le modèle, donc toutes les valeurs pour ces propriétés fournies ici seront remplacées. `SideployVpc` n'est pas`true`Cette propriété sera ignorée.
TopicArnEnvironmentVariableName ?	`string`	Nom facultatif pour la variable d'environnement ARN de rubrique SNS définie pour la fonction Lambda.
topicNameEnvironmentaireNomVariableName ?	`string`	Nom facultatif pour la variable d'environnement de nom de rubrique SNS définie pour la fonction Lambda.

Propriétés du modèle

Nom	Type	Description
LambdaFonction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.

Nom	Type	Description
snsTopic	`sns.Topic`	Renvoie une instance de la rubrique SNS créée par le modèle.
VPC ?	`ec2.IVpc`	Renvoie une instance du VPC utilisé par le modèle (le cas échéant). Il peut s'agir d'un VPC créé par le modèle ou du VPC fourni au constructeur de modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de la construction sans remplacement définira les valeurs par défaut suivantes :

Fonction AWS Lambda

- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Activez le suivi X-Ray.
- Définissez les variables d'environnement :
  - `SNS_TOPIC_NAME` (default)
  - `SNS_TOPIC_ARN` (default)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Rubrique Amazon SNS

- Configurez les autorisations d'accès les moins privilèges pour la rubrique SNS.
- Activez le chiffrement côté serveur à l'aide d'une clé KMS gérée par AWS.
- Application du chiffrement des données en transit.

Architecture

GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructions/aws-lambda-sns

aws-lambda-sqs

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion de versions sémantiquesModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

AWS Solutions Construct implémente une fonction AWS Lambda connectée à une file d'attente Amazon SQS.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`LambdaToSqsProps`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Fonction Lambda optionnelle existante à utiliser à la place de la fonction par défaut. Fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda.
QueueObj existant ?	`sqs.Queue`	Une file d'attente SQS existante facultative à utiliser à la place de la file d'attente par défaut. Fournissant à la fois ceci et`queueProps` provoquera une erreur.
QueueProps ?	`sqs.QueueProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la file d'attente SQS.
EnableQueueUrging ?	`boolean`	Indique s'il faut accorder des autorisations supplémen

Nom	Type	Description
		taires à la fonction Lambda lui permettant de purger la file d'attente SQS. La valeur par défaut est `false`.
Déploiement Deadlette rQueue ?	`boolean`	Indique si une file d'attente secondaire doit être utilisée comme file d'attente de lettres mortes. La valeur par défaut est `true`.
DeadletterQueueProps ?	`sqs.QueueProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de la file d'attente de lettres mortes. Utilisé uniquement si le`deployDeadLetterQueue` a la valeur true.
MaxReceiveCount ?	`number`	Nombre de fois qu'un message peut être retiré sans succès avant d'être déplacé vers la file d'attente de lettres mortes. La valeur par défaut est `15`.

Nom	Type	Description
VPC existant ?	`ec2.IVpc`	Un VPC existant optionnel dans lequel ce modèle doit être déployé. Lorsqu'elle est déployée dans un VPC, la fonction Lambda utilise les ENI du VPC pour accéder aux ressources réseau et un point de terminais on d'interface est créé dans le VPC pour Amazon SQS. Si un VPC existant est fourni, le`deployVpc` ne peut pas être`true`. Une`ec2.IVpc`est utilisé pour permettre aux clients de fournir des VPC qui existent en dehors de la pile à l'aide de la méthode`ec2.Vpc.fromLookup()` Méthode.

Nom	Type	Description
Déploiement de VPC ?	`boolean`	Que ce soit pour créer un nouveau VPC basé sur`vpcProps`dans lequel déployer ce modèle. Paramètre sur`true`déploiera le VPC minimal et le plus privé pour exécuter le modèle : • Un sous-réseau isolé dans chaque zone de disponibilité utilisée par le programme CDK • `enableDnsHostnames` and`enableDns Support` seront tous les deux réglés sur`true` Si cette propriété est`true`, puis`existingVpc` ne peut pas être spécifié. La valeur par défaut est `false`.

Nom	Type	Description
VPCProps ?	`ec2.VpcProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du nouveau VPC.`enableDnsHostnames` ,`enableDnsSupport` ,`natGateways` , `etsubnetConfiguration` sont définies par le modèle, donc toutes les valeurs de ces propriétés fournies ici seront remplacées. Si`deployVpc` n'est pas`true`, cette propriété sera ignorée.
QueueEnvironmentVariableName ?	`string`	Nom facultatif de la variable d'environnement d'URL de file d'attente SQS définie pour la fonction Lambda.

Propriétés du modèle

Nom	Type	Description
DeadletterQueue ?	`sqs.Queue`	Renvoie une instance de la file d'attente de lettres mortes créée par le modèle, si une instance est déployée.
LambdaFonction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.

Nom	Type	Description
SQSqueue	`sqs.Queue`	Renvoie une instance de la file d'attente SQS créée par le modèle.
VPC ?	`ec2.IVpc`	Renvoie une instance du VPC créée ou utilisée par le modèle (le cas échéant). Il peut s'agir d'un VPC créé par le modèle ou d'un VPC fourni au constructeur de modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de la construction sans remplacement définira les valeurs par défaut suivantes :

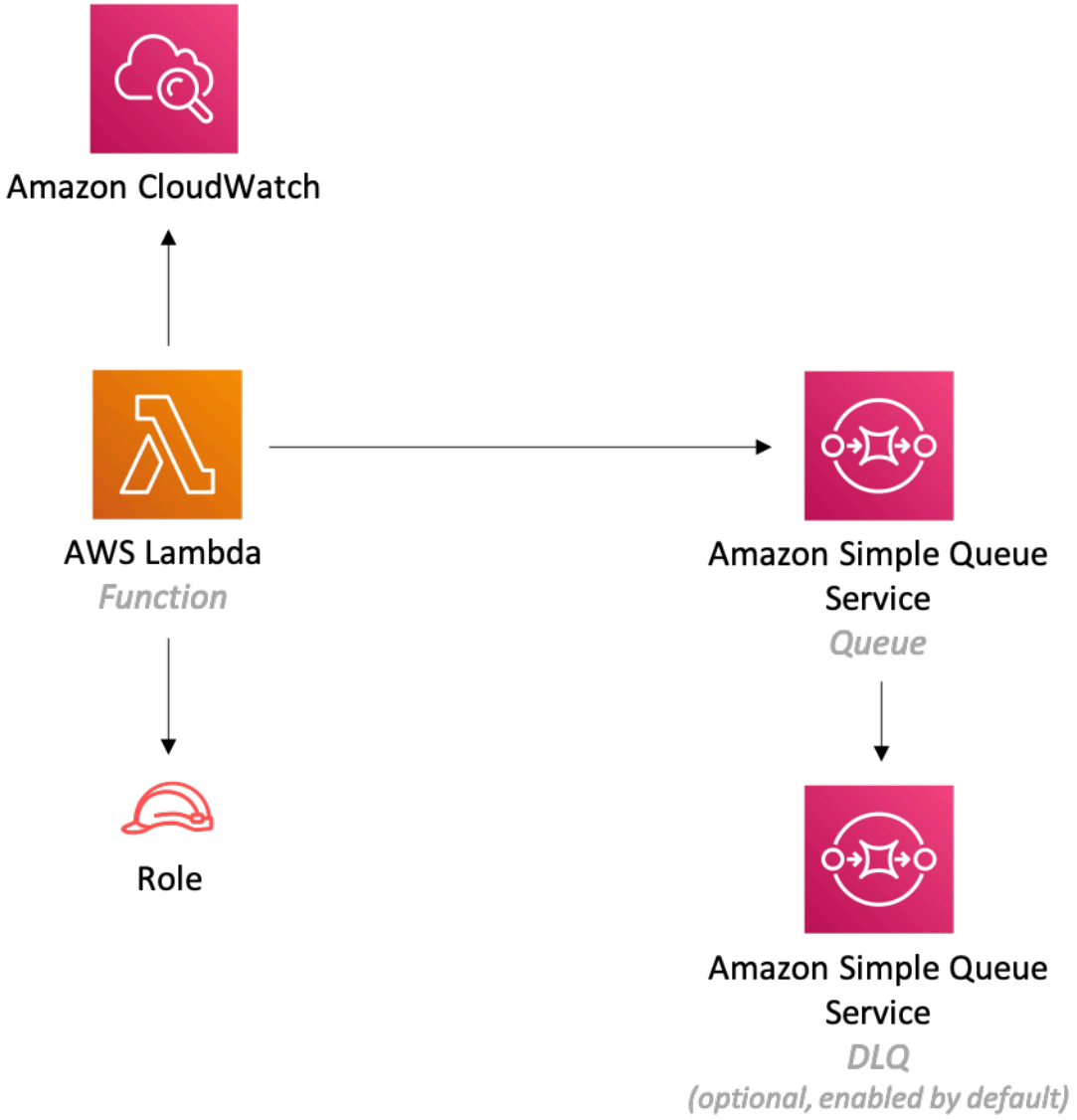
Fonction AWS Lambda

- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Autoriser la fonction à envoyer uniquement des messages à la file d'attente (la purge peut être activée à l'aide de l'option`enableQueuePurge`propriété).
- Activer le suivi X-Ray
- Définissez les variables d'environnement :
  - `SQS_QUEUE_URL`
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

File d'attente Amazon SQS

- Déployez la file d'attente de lettres mortes SQS pour la file d'attente SQS source.
- Activer le chiffrement côté serveur pour la file d'attente source SQS à l'aide de la clé KMS gérée AWS.
- Application du chiffrement des données en transit.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructions/aws-lambda-sqs

aws-lambda-sqs-lambda

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion de versions sémantiquesModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

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

Overview

Ce modèle AWS Solutions Constructs implémente (1) une fonction AWS Lambda configurée pour envoyer des messages à une file d'attente ; (2) une file d'attente Amazon SQS ; et (3) une fonction AWS Lambda configurée pour consommer des messages de la file d'attente.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`LambdaToSqsToLambdaProps`

Modèle de construction

Nom	Type	Description
Producteur existant Lambdaobj ?	`lambda.Function`	Fonction Lambda optionnel le existante à utiliser à la place de la fonction par défaut pour envoyer des messages à la file d'attente. Fournir à la fois ceci et`producerL`

Nom	Type	Description
		`ambdaFunctionProps` provoquera une erreur.
ProducerLambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda du producteur.
QueueObj existant ?	`sqs.Queue`	Une file d'attente SQS existante facultative à utiliser à la place de la file d'attente par défaut. Fournir à la fois ceci et`queueProps` provoquera une erreur.
QueueProps ?	`sqs.QueueProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la file d'attente SQS. Fournir à la fois ceci et`existingQueueObj` provoquera une erreur.
Déploiement DeadletterQueue ?	`boolean`	Indique s'il faut créer une file d'attente secondaire à utiliser comme file d'attente de lettres mortes. La valeur par défaut est `true`.

Nom	Type	Description
DeadletterQueueProps ?	`sqs.QueueProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de la file d'attente de lettres mortes. Utilisé uniquement si le`deployDeadLetterQueue` est définie sur`true`.
MaxReceiveCount ?	`number`	Nombre de fois qu'un message peut être déplacé sans succès avant d'être déplacé vers la file d'attente de lettres mortes. La valeur par défaut est `15`.
ExistantConsumerLambdaobj ?	`lambda.Function`	Fonction Lambda optionnelle existante à utiliser à la place de la fonction par défaut pour réception/consommer des messages de la file d'attente. Fournir à la fois ceci et`consumerLambdaFunctionProps` provoquera une erreur.
ConsumerLambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda consommateur.

Nom	Type	Description
QueueEnvironmentVariableName ?	`string`	Nom facultatif de la variable d'environnement d'URL de file d'attente SQS définie pour la fonction Lambda du producteur.

Propriétés du modèle

Nom	Type	Description
Consommateur Lambdafunction	`lambda.Function`	Renvoie une instance de la fonction Lambda consommateur créée par le modèle.
DeadletterQueue ?	`sqs.Queue`	Renvoie une instance de la file d'attente de lettres mortes créée par le modèle, si une instance est déployée.
ProducteurLambdafunction	`lambda.Function`	Renvoie une instance de la fonction Lambda producteur créée par le modèle.
SQSqueue	`sqs.Queue`	Renvoie une instance de la file d'attente SQS créée par le modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de cette construction (sans propriétés remplacées) respectera les valeurs par défaut suivantes :

Fonctions AWS Lambda

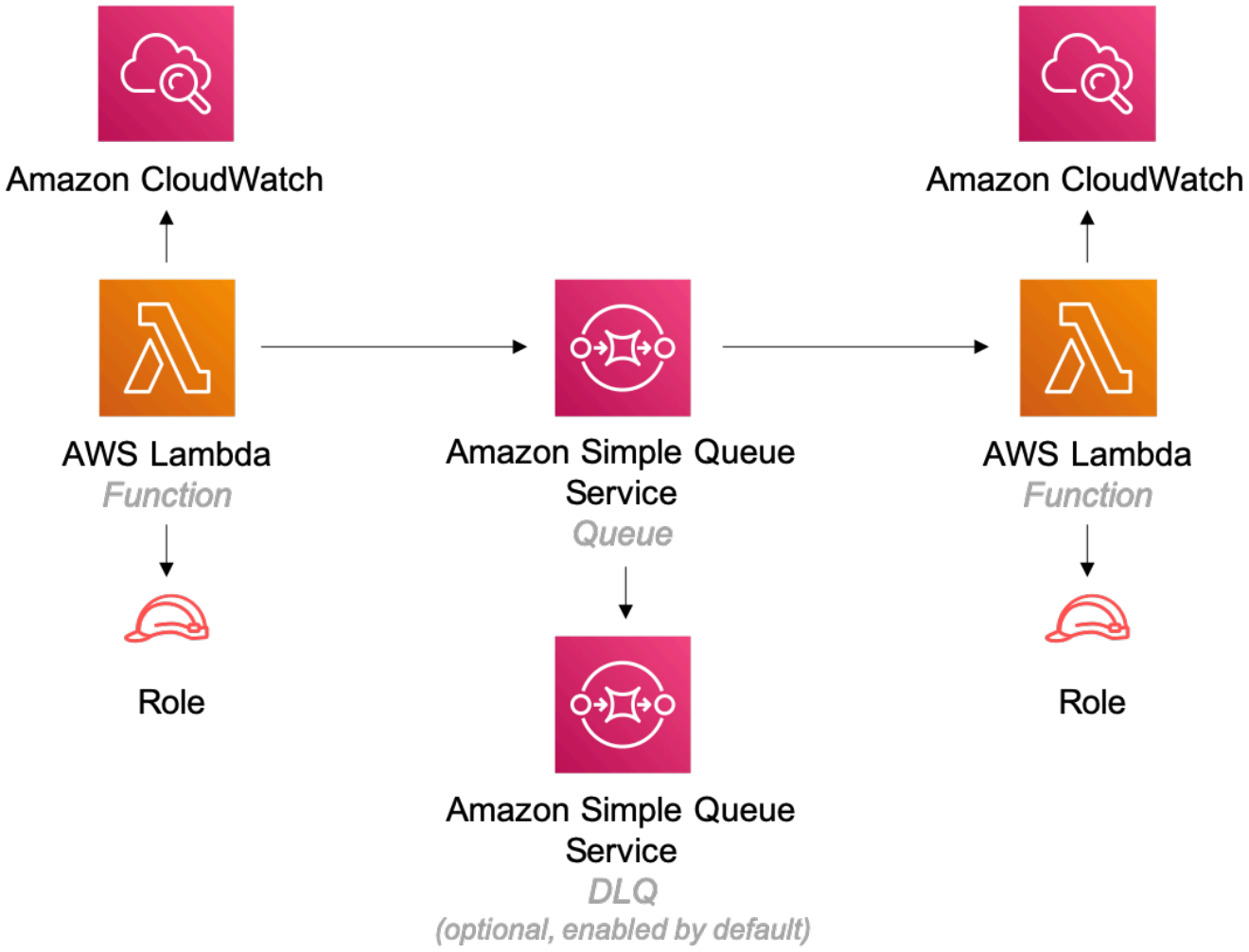
- Configurer le rôle IAM d'accès limité aux privilèges pour les fonctions Lambda.

- Activez la réutilisation des connexions avec les fonctions Keep-Alive pour NodeJS Lambda.
- Activer le suivi X-Ray
- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

File d'attente Amazon SQS

- Déployer une file d'attente de lettre morte pour la file d'attente principale.
- Activer le chiffrement côté serveur pour la file d'attente principale à l'aide d'une clé AWS Managed KMS.
- Application du chiffrement des données en transit

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-constructions/aws-lambda-sqs-lambda

aws-lambda-step-function

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Ceux-ci ne sont pas assujettis à laGestion de version sémantiqueModèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_lambda_step_function`
TypeScript	`@aws-solutions-constructs/aws-lambda-step-function`
Java	`software.amazon.awsconstructs.services.lambdastepfunction`

Overview

Cette solution AWS Construct implémente une fonction AWS Lambda connectée à une fonction AWS Step.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`LambdaToStepFunctionProps`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et `lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un `existingLambdaObj` est fourni.
StateMachineProps	`sfn.StateMachineProps`	Les accessoires fournis par l'utilisateur pour SFN.State Machine.
CreateCloudWatchArms	`boolean`	Indique s'il faut créer des alarmes CloudWatch recommandées.
LogGroupProps ?	`logs.LogGroupProps`	Accessoires fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour le groupe de journaux CloudWatch Logs.
StateMachineEnvironmentVariableName	`string`	Nom facultatif pour la variable d'environnement de machine d'état Step Functions définie pour la fonction Lambda du producteur.

Propriétés de modèle

Nom	Type	Description
Cloudwatch Alarm ?	`cloudwatch.Alarm[]`	Renvoie une liste d'une ou plusieurs alarmes CloudWatch créées par le modèle.
LambdaFunction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.
StateMachine	`sfn.StateMachine`	Renvoie une instance de la machine d'état créée par le modèle.
StateMachineLogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle pour la machine d'état.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

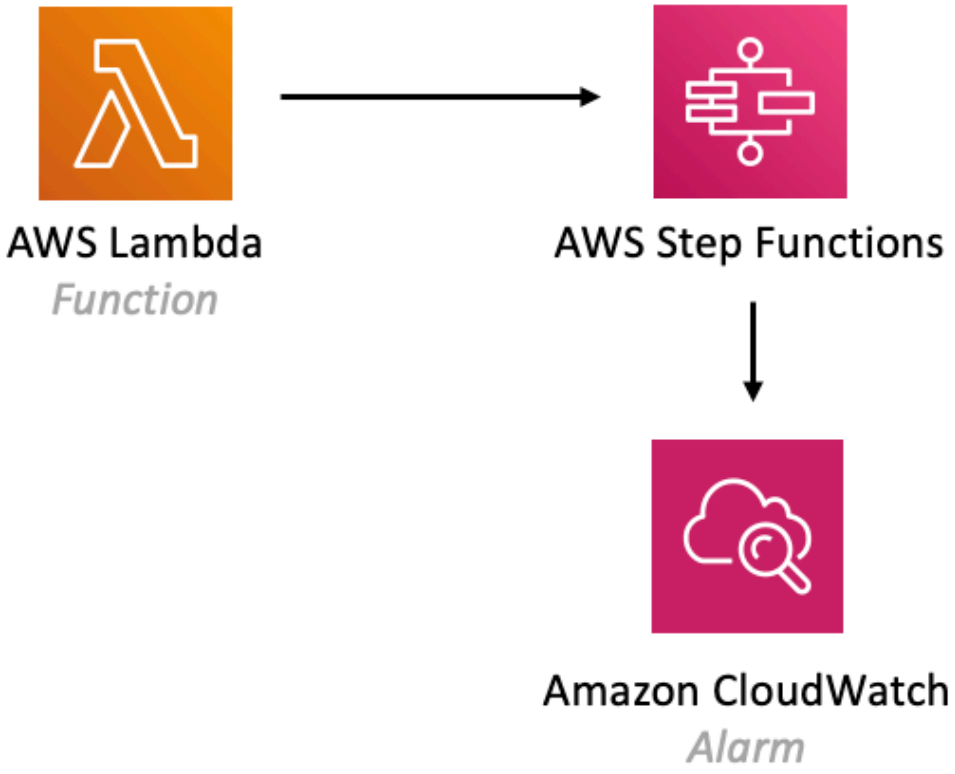
Fonction AWS Lambda

- Configurez un rôle IAM à accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec les fonctions Keep-Alive pour NodeJS Lambda.
- Activer le suivi X-Ray.
- Définir les variables d'environnement :
  - `STATE_MACHINE_ARN` (default)
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

AWS Step Functions

- Déployez les alarmes CloudWatch les meilleures pratiques pour la machine d'état AWS Step Functions.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructions/aws-lambda-step-function

aws-s3-lambda

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion sémantique des versionsLe modèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_s3_lambda`
TypeScript	`@aws-solutions-constructs/aws-s3-lambda`
Java	`software.amazon.awsconstructs.services.s3lambda`

Overview

Ce module AWS Solutions Construct implémente un compartiment Amazon S3 connecté à une fonction AWS Lambda.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`S3ToLambdaProps`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFunctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si un`existingLambdaObj` est fourni.

Nom	Type	Description
Bucketobj existant ?	`s3.Bucket`	Instance existante de l'objet S3 Bucket. Si cela est fourni, alors fournir également `bucketProps` est une erreur.
BucketProps ?	`s3.BucketProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du compartiment. Ignoré si `unexistingBucketObj` est fourni.
S3EventSourceProps ?	`S3EventSourceProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut pour S3EventSourceProps

Propriétés du modèle

Nom	Type	Description
LambdaBonction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.
S3Bucket ?	`s3.Bucket`	Renvoie une instance du compartiment S3 créé par le modèle.
S3LoggingBucket ?	`s3.Bucket`	Renvoie une instance du compartiment de journalisation créé par le modèle pour le compartiment S3.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

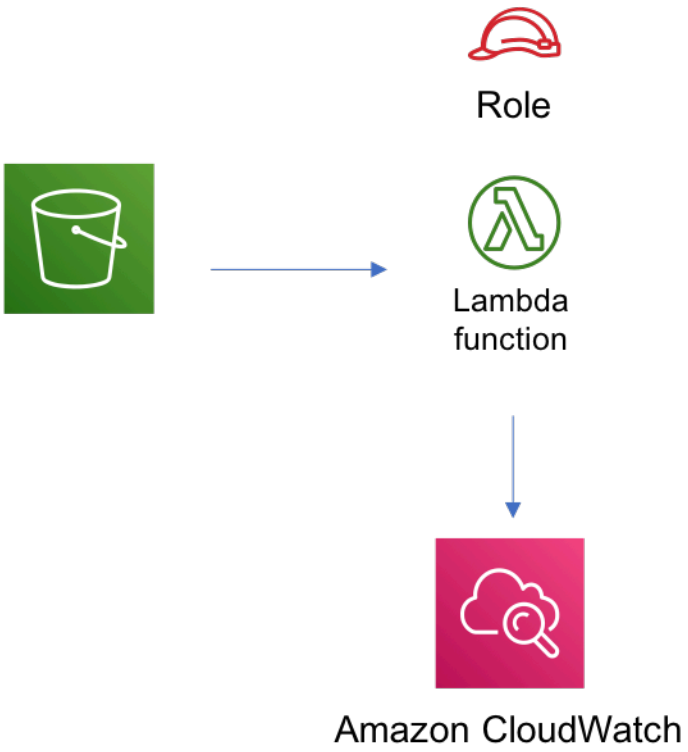
Bucket Amazon S3

- Configurez la journalisation d'accès pour le compartiment S3.
- Activez le chiffrement côté serveur pour le compartiment S3 à l'aide de la clé KMS gérée par AWS.
- Activez le contrôle de version pour S3 Bucket.
- N'autorisez pas l'accès public pour le compartiment S3.
- Conservez le compartiment S3 lors de la suppression de la pile CloudFormation.
- Application du chiffrement des données en transit.
- Applique la règle de cycle de vie pour déplacer les versions d'objets non actuelles vers le stockage Glacier après 90 jours.

Fonction AWS Lambda

- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Activer le suivi X-Ray.
- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-construction/aws-s3-lambda

aws-s3-sqs

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion de versions sémantiquesmodèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_s3_sqs`
TypeScript	`@aws-solutions-constructs/aws-s3-sqs`
Java	`software.amazon.awsconstructs.services.s3sqs`

Overview

Ce module AWS Solutions Construct implémente un compartiment Amazon S3 configuré pour envoyer des notifications à une file d'attente Amazon SQS.

Voici une définition de modèle déployable minimale dans TypeScript :

```
import { S3ToSqs } from "@aws-solutions-constructs/aws-s3-sqs";

new S3ToSqs(stack, 'S3ToSQSPattern', {});
```

Initializer

```
new S3ToSqs(scope: Construct, id: string, props: S3ToSqsProps);
```

Paramètres

- scope`Construct`

- `idstring`
- propsS3ToSqsProps

Accessoires de construction de modèle

Nom	Type	Description
Bucketobj existant ?	s3.Bucket	Instance existante de l'objet S3 Bucket. Si cela est fourni, alors fournir également `bucketProps` est une erreur.
BucketProps ?	s3.BucketProps	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut du compartiment S3.
S3EventTypes ?	s3.EventType[]	Types d'événements S3 qui déclencheront la notificat ion. La valeur par défaut est `s3.EventType.OBJEC T_CREATED` .
S3EventFilters ?	s3.NotificationKey Filter[]	Les règles de filtre de clé d'objet S3 pour déterminer quels objets déclenchent cet événement. Si ce n'est pas spécifié, aucune règle de filtre ne sera appliquée.
QueueObj existant ?	sqs.Queue	Une file d'attente SQS existante facultative à utiliser à la place de la file d'attente par défaut. Fournir à la fois ceci et`queueProps` provoquera une erreur. Si la file d'attente SQS est chiffrée, la clé KMS

Nom	Type	Description
		utilisée pour le chiffrement doit être un CMK géré par le client.
QueueProps ?	`sqs.QueueProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la file d'attente SQS. Ignoré si `unexistingQueueObj` est fourni.
DeadletterQueueProps ?	`sqs.QueueProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de la file d'attente de lettres mortes. Utilisé uniquement si le`deployDeadLetterQueue` est définie sur true.
Déploiement Deadletter Queue ?	`boolean`	Indique si vous voulez créer une file d'attente secondaire à utiliser comme file d'attente de lettres mortes. La valeur par défaut est `true`.
MaxReceiveCount ?	`number`	Nombre de fois qu'un message peut être retiré sans succès avant d'être déplacé vers la file d'attente de lettres mortes. La valeur par défaut est `15`.

Nom	Type	Description
EnableEncryptionWithCustomerManagedKey ?	`boolean`	Indique s'il faut utiliser une clé KMS, soit gérée par cette application CDK, soit importée. Si vous importez une clé de chiffrement, elle doit être spécifiée dans le champ`encryptionKey` pour cette construction.
encryptionKey	`kms.Key`	Une clé de chiffrement existante facultative à utiliser à la place de la clé de chiffrement par défaut.
EncryptionKeyProps ?	`kms.KeyProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la clé de chiffrement.

Propriétés du modèle

Nom	Type	Description
SQSqueue	`sqs.Queue`	Renvoie une instance de la file d'attente SQS créée par le modèle.
DeadletterQueue ?	`sqs.Queue`	Renvoie une instance de la file d'attente de lettres mortes créée par le modèle, si une instance est déployée.

Nom	Type	Description
encryptionKey	`kms.IKey`	Renvoie une instance de la clé de chiffrement créée par le modèle.
S3Bucket	`s3.Bucket`	Renvoie une instance du compartiment S3 créé par le modèle.
S3LoggingBucket ?	`s3.Bucket`	Renvoie une instance du compartiment de journalisation créé par le modèle pour le compartiment S3.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Bucket Amazon S3

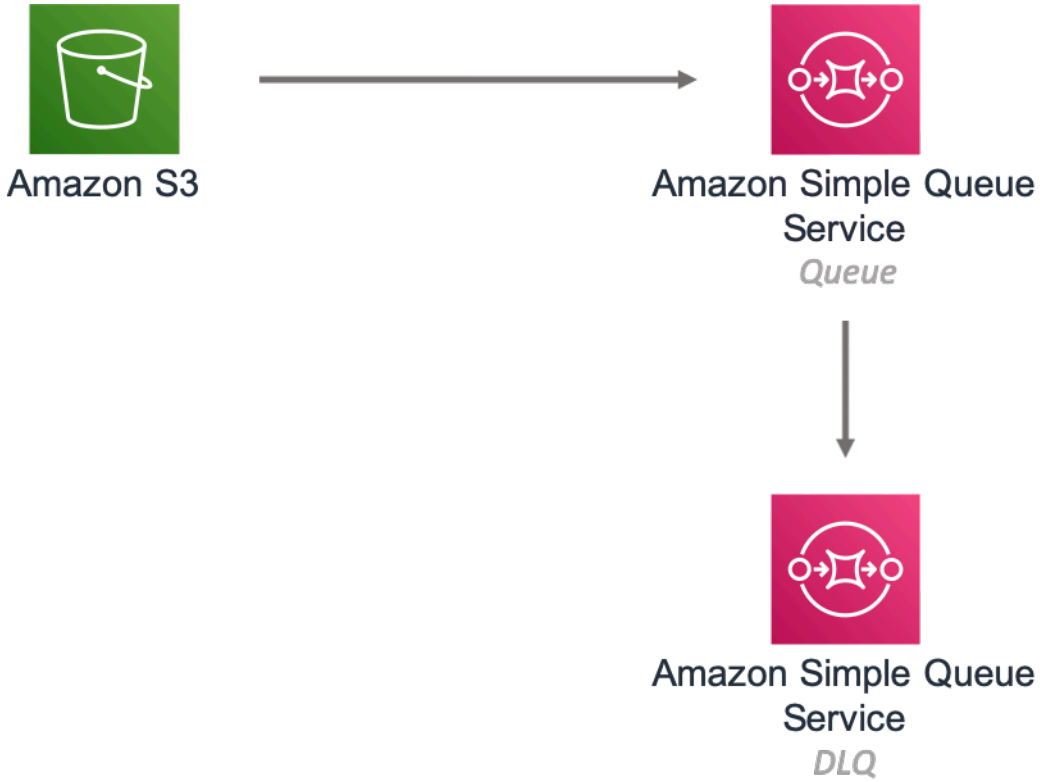
- Configurer la journalisation d'accès pour le compartiment S3
- Activer le chiffrement côté serveur pour le compartiment S3 à l'aide de la clé KMS gérée par AWS
- Activer le contrôle de version pour S3 Bucket
- Ne pas autoriser l'accès public pour le compartiment S3
- Conserver le compartiment S3 lors de la suppression de la pile CloudFormation
- Application du chiffrement des données en transit
- Applique la règle de cycle de vie pour déplacer les versions d'objets non actuelles vers le stockage Glacier après 90 jours

File d'attente Amazon SQS

- Configurer les autorisations d'accès les moins privilèges pour SQS File d'attente
- Déployer la file d'attente de lettres mortes SQS pour la file d'attente SQS source
- Activer le chiffrement côté serveur pour SQS

- Application du chiffrement des données en transit

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-construction/aws-s3-sqs

aws-s3-step-function

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion de version sémantiqueModèle. Cela signifie que même si vous pouvez les

utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_s3_step_function`
TypeScript	`@aws-solutions-constructs/aws-s3-step-function`
Java	`software.amazon.awsconstructs.services.s3stepfunction`

Overview

Ce module AWS Solutions Construct implémente un compartiment Amazon S3 connecté à une fonction AWS Step.

> **Note**
>
> Cette construction utilise Amazon EventBridge (Amazon CloudWatch Events) pour déclencher AWS Step Functions. EventBridge est plus flexible, mais le déclenchement des Step Functions avec les notifications d'événements S3 a moins de latence et est plus rentable. Si le coût et/ou la latence est un problème, vous devriez envisager de déployer`aws-s3-lambda`and`aws-lambda-stepfunctions`à la place de cette construction.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`S3ToStepFunctionProps`

Accessoires de construction de modèle

Nom	Type	Description
Bucketobj existant ?	`s3.IBucket`	Instance existante de l'objet S3 Bucket. Si cela est fourni, alors fournir également `bucketProps` est une erreur.
BucketProps ?	`s3.BucketProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut du compartiment. Ignoré si `unexistingBucketObj` est fourni.

Nom	Type	Description
StateMachineProps	`sfn.StateMachineProps`	Les accessoires fournis par l'utilisateur facultatif pour remplacer les accessoires par défaut pour SFN.State Machine.
EventTruleProps ?	`events.RuleProps`	L'utilisateur facultatif a fourni EventRuleProps pour remplacer les valeurs par défaut.
Déploiement de CloudTrail ?	`boolean`	Indique s'il faut déployer un Trail dans AWS CloudTrail pour consigner les événement s d'API dans Amazon S3. La valeur par défaut est `true`.
CreateCloudWatchArms	`boolean`	Indique s'il faut créer des alarmes CloudWatch recommandées.
LogGroupProps ?	`logs.LogGroupProps`	Accessoires fournis par l'utilisateur facultatifs pour remplacer les accessoires par défaut du groupe de journaux CloudWatch Logs.

Propriétés du modèle

Nom	Type	Description
Cloudtrail ?	`cloudtrail.Trail`	Renvoie une instance de la piste Cloudtrail créée par le modèle.

Nom	Type	Description
CloudTrailBucket ?	`s3.Bucket`	Renvoie une instance du compartiment créé par le modèle pour stocker les données de trace Cloudtrail.
CloudTrailLoggingBucket ?	`s3.Bucket`	Renvoie une instance du compartiment de journalisation créé par le modèle pour le compartiment principal utilisé par la piste Cloudtrail.
Cloudwatch Alarm ?	`cloudwatch.Alarm[]`	Renvoie la liste d'une ou plusieurs alarmes CloudWatch créées par le modèle.
S3Bucket ?	`s3.Bucket`	Renvoie une instance du compartiment S3 créé par le modèle.
S3LoggingBucket ?	`s3.Bucket`	Renvoie une instance du compartiment de journalisation créé par le modèle pour le compartiment S3.
StateMachine	`sfn.StateMachine`	Renvoie une instance de la machine d'état créée par le modèle.
StateMachineLogGroup	`logs.LogGroup`	Renvoie une instance du groupe de journaux créé par le modèle pour la machine d'état.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

Bucket Amazon S3

- Configurez la journalisation d'accès pour le compartiment S3.
- Activez le chiffrement côté serveur pour le compartiment S3 à l'aide de la clé KMS gérée par AWS.
- Activez le contrôle de version pour S3 Bucket.
- N'autorisez pas l'accès public pour le compartiment S3.
- Conservez le compartiment S3 lors de la suppression de la pile CloudFormation.
- Application du chiffrement des données en transit.
- Applique la règle de cycle de vie pour déplacer les versions d'objets non actuelles vers le stockage Glacier après 90 jours.

AWS CloudTrail

- Configurez un Trail dans AWS CloudTrail pour consigner les événements d'API dans Amazon S3 liés au compartiment créé par le Construct.

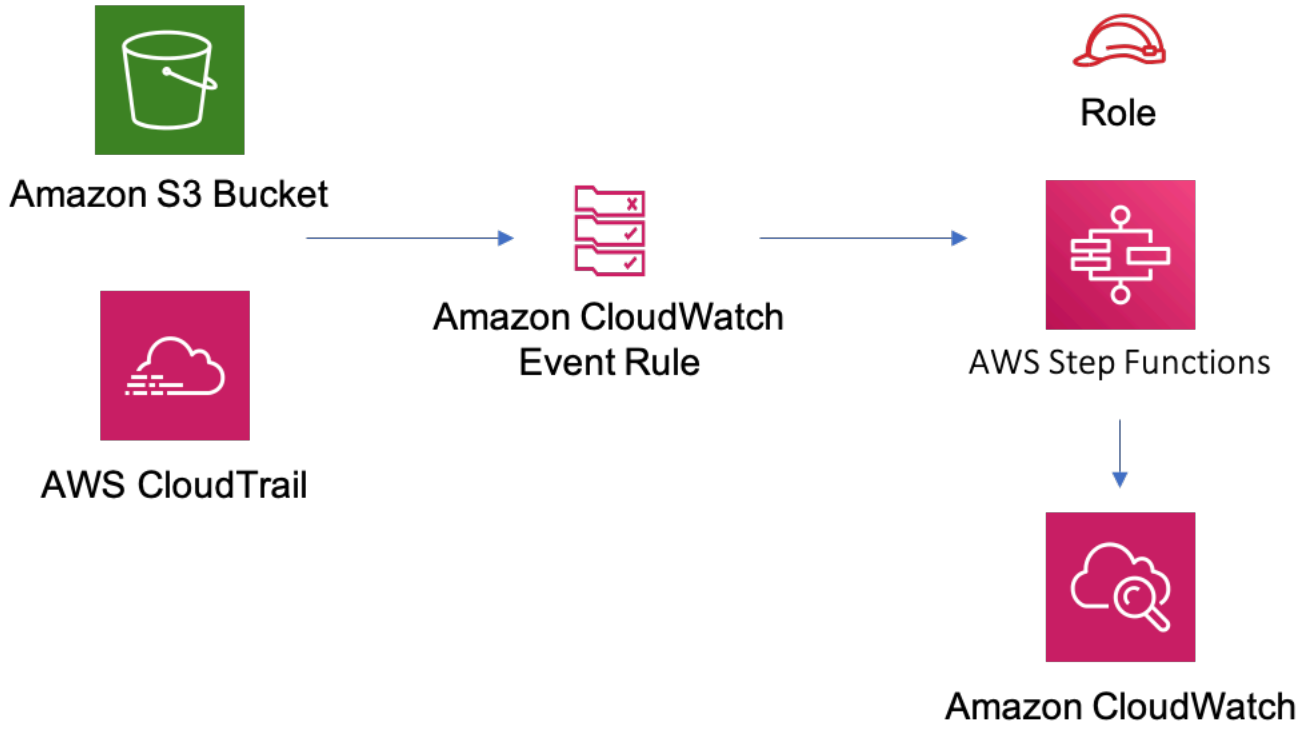
Règle Amazon CloudWatch Events

- Accordez les autorisations les moins privilèges aux événements CloudWatch pour déclencher la fonction Lambda.

AWS Step Functions

- Activez la journalisation CloudWatch pour API Gateway.
- Déployez les meilleures pratiques des alarmes CloudWatch pour la fonction Step.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-constructs/aws-s3-step-function

aws-sns-lambda

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Celles-ci ne sont pas assujetties à laGestion de versions sémantiquesLe modèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_sns_lambda`
TypeScript	`@aws-solutions-constructs/aws-sns-lambda`
Java	`software.amazon.awsconstructs.services.snslambda`

Overview

Cette solution AWS Construct implémente un Amazon SNS connecté à une fonction AWS Lambda.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scopeConstruct
- idstring
- propsSnsToLambdaProps

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	lambda.Function	Instance existante de l'objet Lambda Function, fournissant à la fois ceci etlambdaFunctionProps provoquera une erreur.
LambdaFunctionProps ?	lambda.FunctionProps	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si unexistingLambdaObj est fourni.
ExistantTopicObj ?	sns.Topic	Instance existante de l'objet SNS Topic, fournissant à la fois ceci ettopicProps provoquera une erreur.
SujetProps ?	sns.TopicProps	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la rubrique SNS.

Propriétés de modèle

Nom	Type	Description
LambdaUNction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.
snsTopic	`sns.Topic`	Renvoie une instance de la rubrique SNS créée par le modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

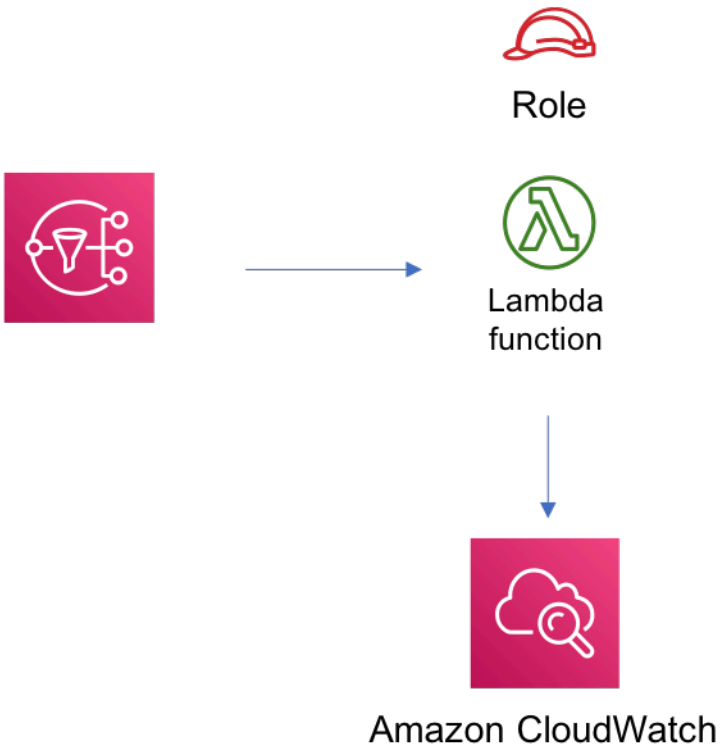
Rubrique Amazon SNS

- Configurez les autorisations d'accès les moins privilèges pour la rubrique SNS.
- Activez le chiffrement côté serveur à l'aide d'une clé KMS gérée par AWS.
- Application du chiffrement des données en transit.

Fonction AWS Lambda

- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Activez le suivi X-Ray.
- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-constructions/aws-sns-lambda

aws-sns-sqs

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion de versions sémantiquesmodèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_sns_sqs`
TypeScript	`@aws-solutions-constructs/aws-sns-sqs`
Java	`software.amazon.awsconstructs.services.snssqs`

Overview

Cette rubrique AWS Solutions Construct implémente une rubrique Amazon SNS connectée à une file d'attente Amazon SQS.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`SnsToSqsProps`

Accessoires de construction de modèle

Nom	Type	Description
ExistantTopicObj ?	`sns.Topic`	Instance existante de l'objet SNS Topic, fournissant à la fois ceci et`topicProps` provoquera une erreur.
SujetProps ?	`sns.TopicProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la rubrique SNS. Ignoré si un`existingTopicObj` est fourni.
QueueObj existant ?	`sqs.Queue`	Une file d'attente SQS existante facultative à utiliser à la place de la file d'attente par défaut. Fournir à la fois ceci et`queueProps` provoquera une erreur.
QueueProps ?	`sqs.QueueProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer

Nom	Type	Description
		les propriétés par défaut de la file d'attente SQS. Ignoré si `unexistingQueueObj` est fourni.
Déploiement DeadletterQueue ?	`boolean`	Indique si vous voulez créer une file d'attente secondaire à utiliser comme file d'attente de lettres mortes. La valeur par défaut est `true`.
DeadletterQueueProps ?	`sqs.QueueProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de la file d'attente de lettres mortes. Utilisé uniquement si le paramètre`deployDeadLetterQueue` est définie sur true.
MaxReceiveCount ?	`number`	Nombre de fois qu'un message peut être défile d'attente sans succès avant d'être déplacé vers la file d'attente de lettres mortes. La valeur par défaut est `15`.
EnableEncryptionWithCustomerManagedKey ?	`boolean`	Indique s'il faut utiliser une clé de chiffrement gérée par le client, soit gérée par cette application CDK, soit importée. Si vous importez une clé de chiffrement, elle doit être spécifiée dans le champ`encryptionKey` pour cette construction.

Nom	Type	Description
encryptionKey ?	`kms.Key`	Une clé de chiffrement existante facultative à utiliser à la place de la clé de chiffrement par défaut.
EncryptionKeyProps ?	`kms.KeyProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la clé de chiffrement.

Propriétés de modèle

Nom	Type	Description
snsTopic	`sns.Topic`	Renvoie une instance de la rubrique SNS créée par le modèle.
encryptionKey	`kms.Key`	Renvoie une instance de la clé de chiffrement créée par le modèle.
SQSqueue	`sqs.Queue`	Renvoie une instance de la file d'attente SQS créée par le modèle.
DeadletterQueue ?	`sqs.Queue`	Renvoie une instance de la file d'attente de lettres mortes créée par le modèle, si une instance est déployée.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

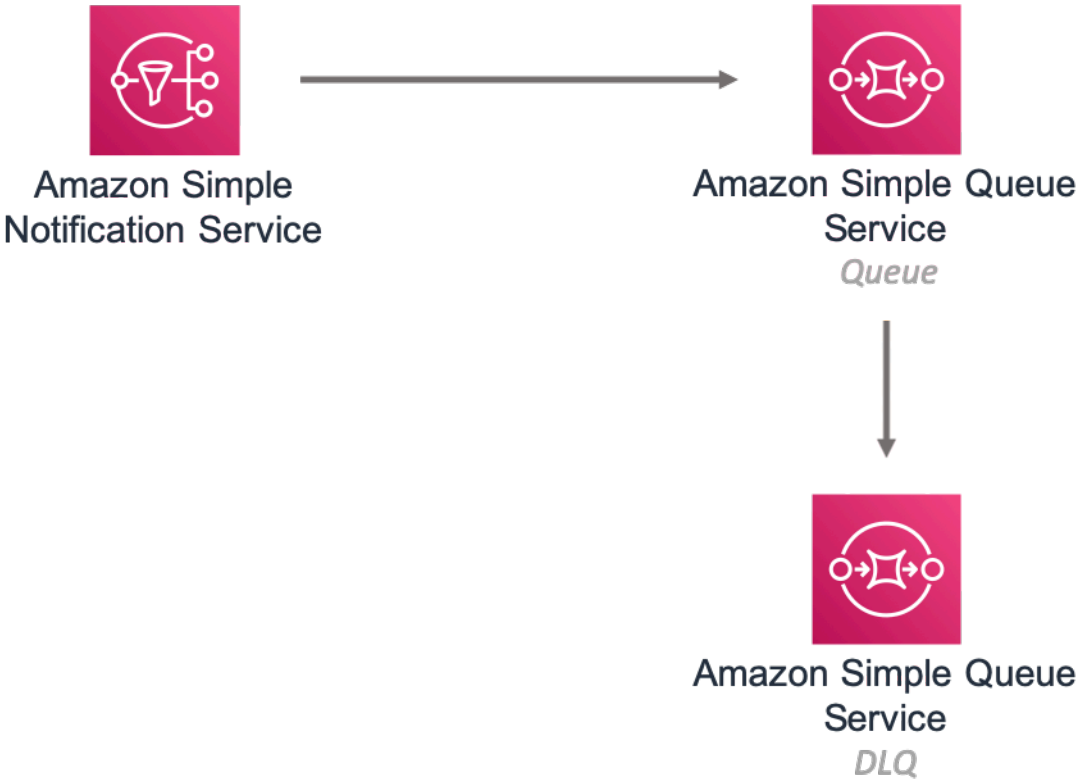
Rubrique Amazon SNS

- Configurez les autorisations d'accès les moins privilèges pour la rubrique SNS.
- Activez le chiffrement côté serveur à l'aide d'une clé KMS gérée par AWS.
- Application du chiffrement des données en transit.

File d'attente Amazon SQS

- Configurer les autorisations d'accès les moins privilèges pour la file d'attente SQS.
- Déployer la file d'attente de lettres mortes pour la file d'attente SQS source.
- Activer le chiffrement côté serveur pour la file d'attente SQS à l'aide d'une clé KMS gérée par le client.
- Application du chiffrement des données en transit.

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :	
	@aws -solutions-constructs/aws-sns-sqs

aws-sqs-lambda

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans n'importe quelle version future. Ceux-ci ne sont pas assujettis à laGestion de versions sémantiquesmodèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

Remarque: Pour garantir une bonne fonctionnalité, les packages AWS Solutions Constructs et AWS CDK de votre projet doivent être la même version.

Langage	Package
Python	`aws_solutions_constructs.aws_sqs_lambda`
TypeScript	`@aws-solutions-constructs/aws-sqs-lambda`
Java	`software.amazon.awsconstructs.services.sqslambda`

Overview

Cette solution AWS Construct implémente une file d'attente Amazon SQS connectée à une fonction AWS Lambda.

Voici une définition de modèle déployable minimale dans 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);
```

Paramètres

- scope`Construct`
- id`string`
- props`SqsToLambdaProps`

Accessoires de construction de modèle

Nom	Type	Description
L'existence de Glambdaobj ?	`lambda.Function`	Instance existante de l'objet Lambda Function, fournissant à la fois ceci et`lambdaFun`

Nom	Type	Description
		`ctionProps` provoquera une erreur.
LambdaFunctionProps ?	`lambda.FunctionProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la fonction Lambda. Ignoré si `unexistingLambdaObj` est fourni.
QueueObj existant ?	`sqs.Queue`	Une file d'attente SQS existante facultative à utiliser à la place de la file d'attente par défaut. Fournir à la fois ceci et`queueProps` provoquera une erreur.
QueueProps ?	`sqs.QueueProps`	Propriétés facultatives fournies par l'utilisateur pour remplacer les propriétés par défaut de la file d'attente SQS. Ignoré si `unexistingQueueObj` est fourni.
Déploiement Deadlette rQueue ?	`boolean`	Indique s'il faut créer une file d'attente secondaire à utiliser comme file d'attente de lettres mortes. La valeur par défaut est `true`.

Nom	Type	Description
DeadletterQueueProps ?	`sqs.QueueProps`	Props fournis par l'utilisateur en option pour remplacer les accessoires par défaut de la file d'attente de lettres mortes. Utilisé uniquement si le`deployDeadLetterQueue` est définie sur true.
MaxReceiveCount ?	`number`	Nombre de fois qu'un message peut être défile d'attente sans succès avant d'être déplacé vers la file d'attente de lettres mortes. La valeur par défaut est `15`.

Propriétés de modèle

Nom	Type	Description
DeadletterQueue ?	`sqs.Queue`	Renvoie une instance de la file d'attente de lettres mortes créée par le modèle, si une instance est déployée.
LambdaUNction	`lambda.Function`	Renvoie une instance de la fonction Lambda créée par le modèle.
SQSqueue	`sqs.Queue`	Renvoie une instance de la file d'attente SQS créée par le modèle.

Paramètres par défaut

L'implémentation prête à l'emploi de ce modèle sans remplacement définira les valeurs par défaut suivantes :

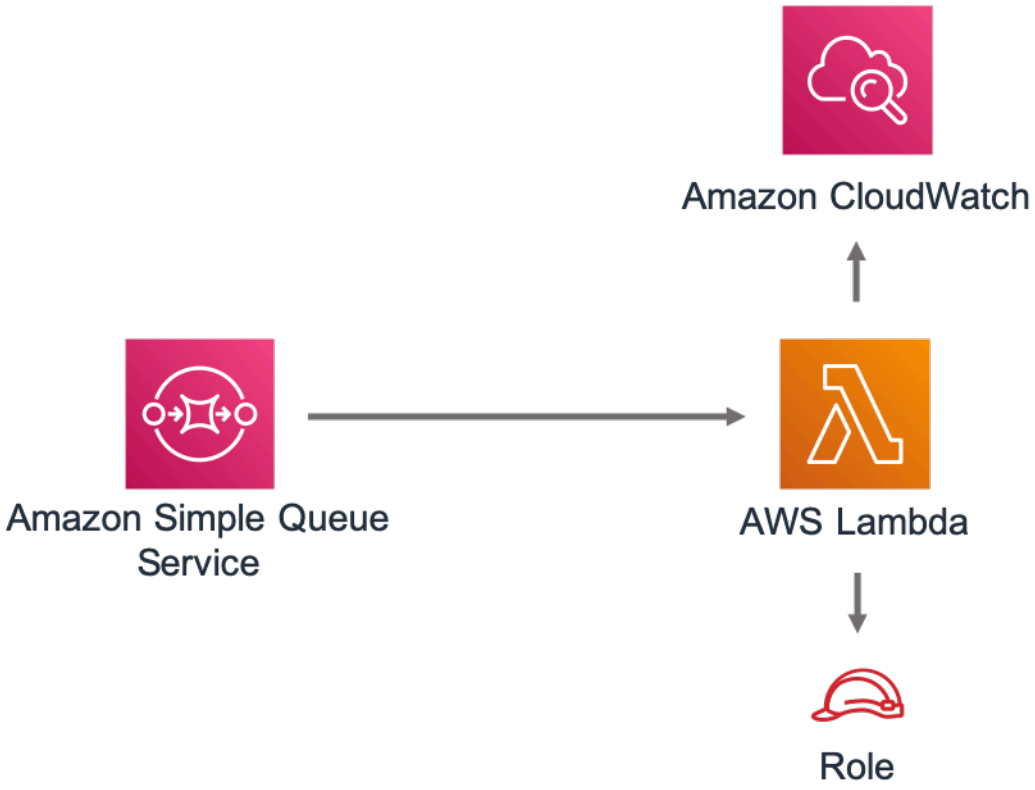
File d'attente Amazon SQS

- Déployez la file d'attente de lettres mortes SQS pour la file d'attente SQS source.
- Activer le chiffrement côté serveur pour la file d'attente SQS source à l'aide de la clé KMS managé AWS.
- Appliquer le chiffrement des données en transit.

Fonction AWS Lambda

- Configurez le rôle IAM d'accès limité pour la fonction Lambda.
- Activez la réutilisation des connexions avec la fonction Keep-Alive pour NodeJS Lambda.
- Activer le suivi X-Ray.
- Définir les variables d'environnement :
  - `AWS_NODEJS_CONNECTION_REUSE_ENABLED`(pour les fonctions Nœud 10.x et supérieures)

Architecture



GitHub

Pour afficher le code de ce modèle, créer/afficher les problèmes et les demandes d'extraction, et plus encore :

	@aws -solutions-construction/aws-sqs-lambda

core

STABILITY EXPERIMENTAL

Toutes les classes sont en cours de développement actif et sujettes à des modifications ou à des suppressions non rétrocompatibles dans toute version future. Ceux-ci ne sont pas assujettis à laGestion des versions sémantiquesLe modèle. Cela signifie que même si vous pouvez les utiliser, vous devrez peut-être mettre à jour votre code source lors de la mise à niveau vers une version plus récente de ce package.

La bibliothèque de base comprend les composants de base des composants AWS Solutions Constructs. Il définit les classes principales qui sont utilisées dans le reste des constructions AWS Solutions.

Propriétés par défaut pour les constructions CDK AWS

La bibliothèque de base définit les propriétés par défaut des constructions AWS CDK utilisées par les constructions AWS Solutions Builts.

Par exemple, ce qui suit est l'extrait de propriétés par défaut pour la construction S3 Bucket créée par la construction AWS Solutions Constructs. Par défaut, il active le chiffrement côté serveur, la gestion des versions du compartiment, bloque tous les accès publics et configure la journalisation des accès S3.

```
{
  encryption: s3.BucketEncryption.S3_MANAGED,
  versioned: true,
  blockPublicAccess: s3.BlockPublicAccess.BLOCK_ALL,
  removalPolicy: RemovalPolicy.RETAIN,
  serverAccessLogsBucket: loggingBucket
}
```

Remplacer les propriétés par défaut

Les propriétés par défaut définies par la bibliothèque Core peuvent être remplacées par les propriétés fournies par l'utilisateur. Par exemple, l'utilisateur peut remplacer la propriété Amazon S3 Block Public Access pour répondre à des exigences spécifiques.

```
  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
  },
});
```

Avertissements de propriété

Lorsqu'une propriété par défaut de la bibliothèque Core est remplacée par une propriété fournie par l'utilisateur, Constructs émet un ou plusieurs messages d'avertissement sur la console mettant en évidence les modifications. Ces messages ont pour but de sensibiliser l'utilisateur à la situation et de prévenir les remplacements involontaires susceptibles de créer des risques pour la sécurité. Ces messages apparaîtront chaque fois que des commandes liées au déploiement/à la construction sont exécutées, y compris`cdk deploy`,`cdk synth`,`npm test`, etc.

Exemple de message :`AWS_CONSTRUCTS_WARNING: An override has been provided for the property: BillingMode. Default value: 'PAY_PER_REQUEST'. You provided: 'PROVISIONED'.`

Activer les avertissements de remplacement

Les messages d'avertissement de remplacement sont activés par défaut, mais peuvent être explicitement activés/désactivés à l'aide de la commande`overrideWarningsEnabled`Variable shell.

- Pour explicitementDésactivation deremplacer les avertissements, exécuter`export overrideWarningsEnabled=false`.
- Pour explicitementActivation d'remplacer les avertissements, exécuter`export overrideWarningsEnabled=true`.
- Pour revenir au paramètre par défaut, exécutez`unset overrideWarningsEnabled`.

Révisions du document

Pour être alerté des mises à jour de AWS Solutions Constructs, abonnez-vous au flux RSS.

update-history-change	update-history-description	update-history-date
Contenu mis à jour	Ajout du modèle aws-lambda-ssmstringparameter. Autres mises à jour mineures de contenu.	27 mai 2021
Contenu mis à jour	Ajout du modèle aws-lambda-secretsmanager. Autres mises à jour mineures de contenu.	12 mai 2021
Contenu mis à jour	Mises à jour des propriétés pour sélectionner les modèles *-lambda. Autres mises à jour mineures de contenu.	17 avril 2021
Contenu mis à jour	Correction d'un problème dans la procédure pas à pas pour les utilisateurs Python et des exemples de propriétés mises à jour pour les constructions contenant des fonctions Lambda.	30 mars 2021
Contenu mis à jour	Corrections mineures et mises à jour des accessoires de modèle et paramètres par défaut pour certains modèles.	8 mars 2021
Contenu mis à jour	Corrections mineures et mises à jour du contenu pas à pas.	4 mars 2021
Contenu mis à jour	Ajout de `.aws-lambda-sagemakerendpoint` et	24 février 2021

	les propriétés mises à jour pour certains modèles Kinesis Firehose.	
Contenu mis à jour	Ajout de `.aws-kinesisstreams-gluejob` et les étapes pas à pas mises à jour pour les utilisateurs Python.	17 février 2021
Contenu mis à jour	Propriétés mises à jour pour`aws-cloudfront-*`Modèles.	9 février 2021
Contenu mis à jour	Ajout d'un lien vers GitHub pour chaque motif.	5 février 2021
Contenu mis à jour	Propriétés mises à jour pour les modèles sélectionnés.	1er février 2021
Contenu mis à jour	Mise à jour de la documentation des propriétés et des paramètres par défaut pour certains modèles.	4 janvier 2021
Contenu mis à jour	Ajout de nouveaux modèles : aws-cloudfront-mediastore et aws-s3-sqs.	20 décembre 2020
Contenu mis à jour	Retrait du motif aws-lambda-sagemaker.	17 novembre 2020
Contenu mis à jour	Ajout de nouveaux modèles : aws-events-rule-kinesisstreams, aws-events-rule-kinesisfirehose-s3 et aws-lambda-sagemaker.	27 octobre 2020

Contenu mis à jour	Mise à jour pour refléter les changements de rupture dans les modèles aws-events-rule-sns et aws-events-rule-sqs : les noms de classe et d'interface ont été modifiés en casse pascal.	22 octobre 2020
Contenu mis à jour	Ajout de modèles aws-apigateway-sagemakerendpoint et aws-kinesisstreams-kinesisfirehose-s3 ; autres mises à jour mineures du contenu existant.	20 octobre 2020
Contenu mis à jour	Ajout du modèle aws-apigateway-iot ; autres mises à jour mineures du contenu existant.	7 octobre 2020
Contenu mis à jour	Mise à jour des extraits de code de motif déployables minimaux et des meilleures pratiques par défaut pour tous les modèles.	5 octobre 2020
Contenu mis à jour	Propriétés mises à jour du modèle aws-kinesisstreams-lambda pour refléter le changement de rupture.	14 septembre 2020
Contenu mis à jour	Correction mineure à la deuxième partie de la procédure pas à pas.	10 septembre 2020
Contenu mis à jour	Ajout de modèles aws-apigateway-kinesisstreams, aws-events-rule-sns et aws-events-rule-sqs.	10 septembre 2020

Contenu mis à jour	Ajout du modèle aws-sns-sqs ; mises à jour de tous les modèles SNS ; corrections typographiques mineures.	2 septembre 2020
Contenu mis à jour	Correction des noms de module pour le modèle aws-sqs-lambda.	31 août 2020
Contenu mis à jour	Correction du nom du module Python pour le modèle aws-dynamodb-stream-lambda-elasticsearch-kibana.	31 août 2020
Contenu mis à jour	Mise à jour des paramètres par défaut pour les modèles Lambda ; autres mises à jour mineures.	27 août 2020
Contenu mis à jour	Propriétés publiques mises à jour pour les modèles S3 ; mises à jour des paramètres par défaut pour les modèles DynamoDB.	10 août 2020
Contenu mis à jour	Mise à jour de plusieurs modèles pour mettre en évidence l'application par défaut du chiffrement en transit.	4 août 2020

Contenu mis à jour	Ajout du modèle aws-lambda-sqs-lambda ; amélioration des instructions de configuration dans le guide de démarrage ; mise à jour de tous les modèles pour rendre des ressources supplémentaires disponibles via les propriétés publiques.	27 juillet 2020
Contenu mis à jour	Ajout du modèle aws-lambda-sqs ; autres mises à jour mineures.	20 juillet 2020
Contenu mis à jour	Suppression des propriétés DeployIAmbda et DeployBucket des modèles pertinents ; autres mises à jour mineures.	9 juillet 2020
Contenu mis à jour	Ajout du modèle aws-lambda-step-function et correction d'erreurs typographiques mineures.	7 juillet 2020
Contenu mis à jour	Ajout de de à. TableObj ? pour sélectionner des modèles DynamoDB.	25 juin 2020
Contenu mis à jour	Plusieurs corrections de texte et corrections pour les liens rompus.	23 Juin 2020
Première version	Constructions de solutions AWS mises à la disposition du public.	22 juin 2020

Notices

Les clients sont responsables de faire leur propre évaluation indépendante des informations contenues dans ce document. Ce document : (a) est fourni à titre informatif uniquement, (b) représente les offres et les pratiques actuelles de produits AWS, qui peuvent être modifiées sans préavis, et (c) ne crée aucun engagement ou assurance de la part d'AWS et de ses sociétés affiliées, fournisseurs ou concédants de licence. Les produits ou services AWS sont fournis « tels quels » sans garantie, déclaration ou condition d'aucune sorte, expresse ou implicite. Les responsabilités et obligations d'AWS vis-à-vis de ses clients sont régies par les contrats AWS. Le présent document ne fait partie d'aucun, et ne modifie aucun, contrat entre AWS et ses clients.

© 2020 Amazon Web Services, Inc. ou ses sociétés apparentées. Tous droits réservés.

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.