

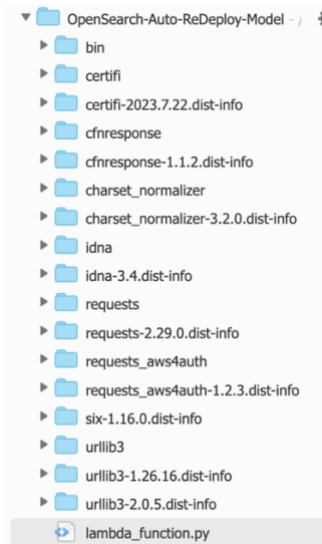
README.pdf

1. Purpose:

This is a lambda helpful function to help the customers to conduct auto-deploy model when the models are undeployed in a node, for example, when adding a new node and the model is not deployed to the new node yet. This helpful lambda function can be added with a trigger to run auto deployment in a schedule.

2. About the zip file:

In the zip file, please note that the `lambda_function.py` is the main file to run in the lambda job, the other folders are imported packages. Those are dependencies for the `lambda_function.py` to run successfully.



3. Set-up Steps by Steps:

3.1 Create [IAM role](#) to give lambda access to OpenSearch

3.1.1 Use the following Custom trust policy to create an AWS IAM Role

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

3.1.2 Add AmazonOpenSearchServiceFullAccess

Permissions policies (1/923) Info

Choose one or more policies to attach to your new role.

Filter by Type			
<input type="text" value="opensearch"/>	<input type="button" value="X"/>	All types	6 matches
Policy name	Type	Description	
<input checked="" type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	AmazonOpenSearchIngestionFullAccess AmazonOpenSearchIngestionReadOnlyAccess AmazonOpenSearchServiceCognitoAccess AmazonOpenSearchServiceFullAccess AmazonOpenSearchServiceReadOnlyAccess AWSQuicksightOpenSearchPolicy	AWS managed AWS managed AWS managed AWS managed AWS managed AWS managed	Allows Amazon OpenSearch Ingestion ... Provides read only access to the Amaz... Provides access to the Amazon Cognit... Provides full access to the Amazon Op... Provides read-only access to the Amaz... Provides access to Amazon OpenSearch...

- Set permissions boundary - *optional*

Cancel

[Previous](#)

Next

IAM > Roles > Create role

Step 1: Select trusted entity
Step 2: Add permissions
Step 3: Name, review, and create

Name, review, and create

Role details

Role name
Enter a meaningful name to identify this role.
test-run-lambda
Maximum 64 characters. Use alphanumeric and "+, @, _" characters.

Description
Add a short explanation for this role.
test-run-lambda
Maximum 1000 characters. Use alphanumeric and "+, @, _" characters.

Step 1: Select trusted entities [Edit](#)

Trust policy

```

1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Principal": {
7         "Service": "lambda.amazonaws.com"
8       },
9       "Action": "sts:AssumeRole"
10    }
11  ]
12 }
```

Step 2: Add permissions [Edit](#)

Permissions policy summary

Policy name	Type	Attached as
AmazonOpenSearchIngestionFullAccess	AWS managed	Permissions policy

3.1.3 After creating the new IAM role, please save the role ARN for later config.

IAM > Roles > LambdaInvokeOpenSearchMLCommonsRole

LambdaInvokeOpenSearchMLCommonsRole [Info](#) [Delete](#)

Role for Lambda to invoke OpenSearch

Summary [Edit](#)

Creation date
November 02, 2023, 18:44 (UTC-07:00)

Last activity
4 days ago

ARN
arn:aws:iam::419213735998:role/LambdaInvokeOpenSearchMLCommonsRole

Maximum session duration
1 hour

[Permissions](#) [Trust relationships](#) [Tags](#) [Access Advisor](#) [Revoke sessions](#)

3.1.4 Map role to backend role with all_access

Navigate to the OpenSearch Dashboard -> Security -> Roles, find all_access role, click on all_access. Navigate to Mapped users -> Managed Mappings

OpenSearch Dashboards

Security **Roles**

Roles (1)

Roles are the core way of controlling access to your cluster. Roles contain any cc tenants. Then you map users to these roles so that users gain those permissions.

all_access

Role	Cluster permissions	Index permissions
all_access	*	*

Rows per page: 10

Map the admin role with the new IAM role created in 3.1.2 step.

Security Roles all_access **Map user**

Map user

Map users to this role to inherit role permissions. Two types of users are supported: user, and backend role. [Learn more](#)

Users

You can create an internal user in internal user database of the security plugin. An internal user can have its own backend role and host for an external authentication and authorization. External users from your identity provider are also supported. [Learn more](#)

Users [Create new internal user](#)

Look up by user name. You can also create new internal user or enter external user.

Backend roles

Use a backend role to directly map to roles through an external authentication system. [Learn more](#)

Backend roles [Remove](#)

[Add another backend role](#)

[Cancel](#) [Map](#)

3.2 Create a new AWS [lambda](#) function:

In 'create function' config, choose RunTime as Python 3.8 and choose use existing role, click on the role name that you created previous in 3.1.2, leave the rest of the default setting, then click "Create Function"

☒ Author from scratch Start with a simple Hello World example.

☐ Use a blueprint Build a Lambda application from sample code and configuration presets for common use cases.

☐ Container image Select a container image to deploy for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Architecture [Info](#)
Choose the instruction set architecture you want for your function code.
☒ x86_64
☐ arm64

Permissions [Info](#)
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

☐ Create a new role with basic Lambda permissions

☒ Use an existing role

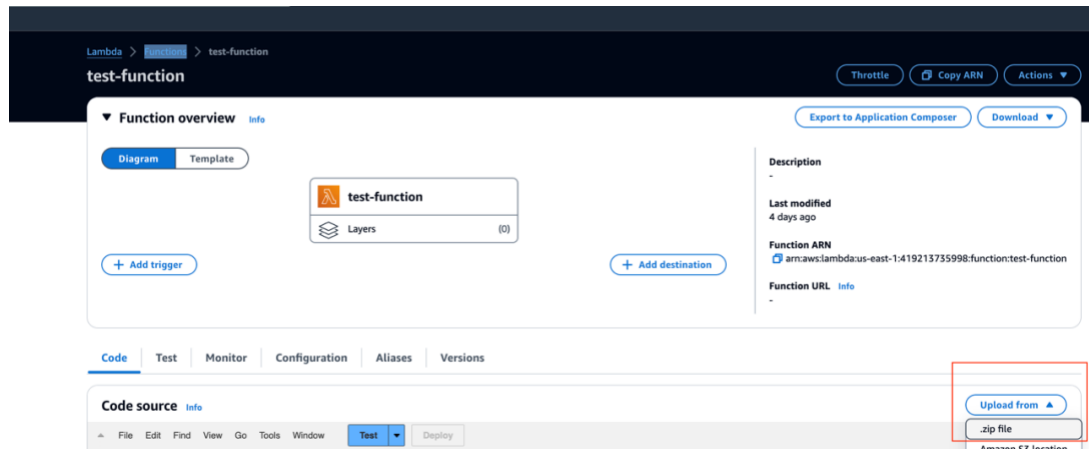
☐ Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

AmazonOpenSearchMLBackendRole

3.3 Upload the zipfile

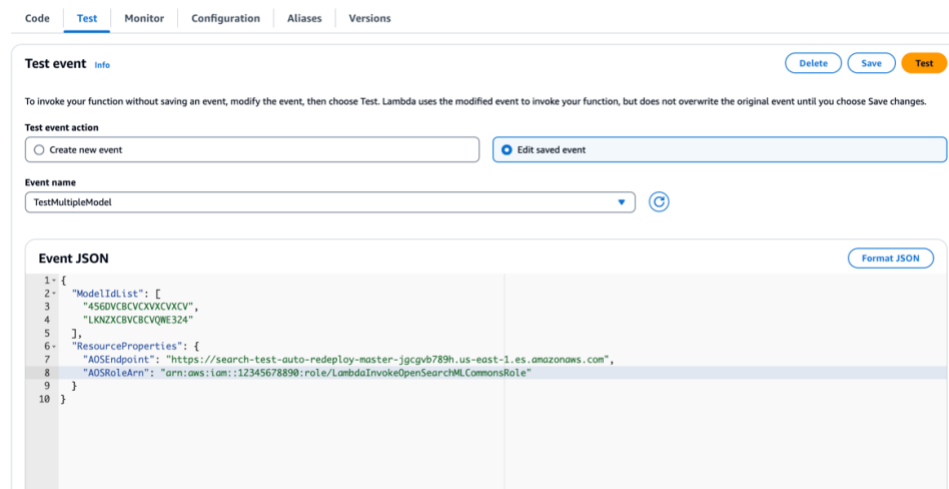
On the new function page, click 'Upload from' in the Code Tap, choose the provided zip file.



3.4 Testing

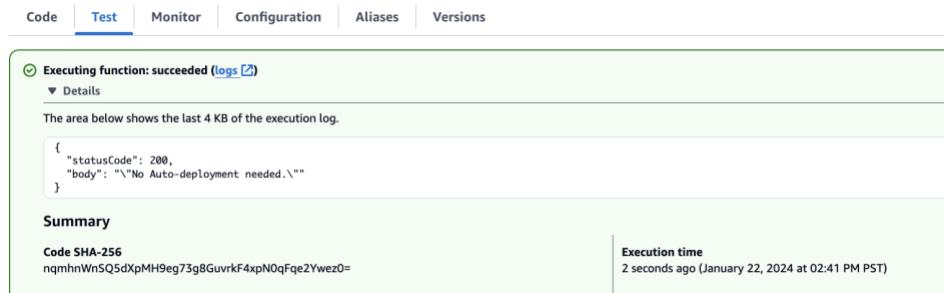
Now you can see that the `lambda_function.py` in the Code source window. Click on Test Tab. Please put the `model_id` into `ModelIdList` that you would like to conduct auto model deployment, input the AOS endpoint which you can find out from AOS domain config and input the lambda role arn that you created in previous 3.1.2. Click “Test” to run auto model deployment.

```
{
  "ModelIdList": [
    "<model_id1>",
    "<model_id2>",
  ],
  "ResourceProperties": {
    "AOSEndpoint": "<AOSEndPoint>",
    "AOSRoleArn": "<RoleARN>"
  }
}
```

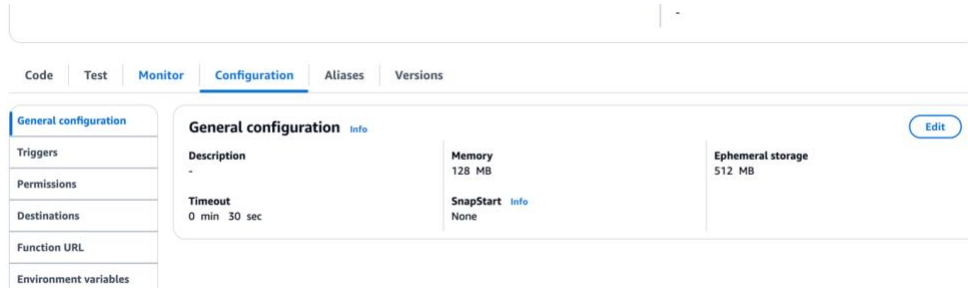


3.4.1 Test Success

Please make sure the test success before adding trigger. The sample success outcome is similar to this.



3.4.2 If lambda timeout, set Timeout to longer timeframe, maximum can be 15 minutes.



3.5 Add EventBridge rule to schedule the lambda job

Open the Amazon EventBridge console. In the navigation pane, choose Rules. Choose Create rule. For Event bus, choose default event bus. For Rule type, choose Schedule. Choose Next. For Schedule pattern, choose a schedule that runs at a regular rate, such as every 10 minutes (0/10 * ? * MON-FRI *). Please refer to the [Cron expressions reference](#) to config different schedules. Choose Next. For Target types, choose AWS service. For Select a target, choose Lambda function from the drop-down list, choose the same lambda job that you created in step 3.2.

In payload, input the same event that you tested in 3.4.1, this payload will pass to lambda job in every scheduled trigger. Review the details of the rule and choose Create rule.

```
{
  "ModelIdList": [
    "<model_id>"
  ],
  "ResourceProperties": {
    "AOSEndpoint": "<AOSEndPoint>",
    "AOSRoleArn": "<RoleARN>"
  }
}
```

Invoke Universal target definition

AWS Lambda

Lambda function

Select ↕ ↻ Create new Lambda function ↗

► **Configure version/alias**

Payload
The JSON that you want to provide to your Lambda function as input. For example, `--payload { key: value }`. [Learn more](#) ↗

1

JSON Ln 1, Col 1 ⓘ ⚠ Errors: 0 ⚠ Warnings: 0 ⚙

Cancel Skip to Review and create schedule Previous Next

Now, the auto deployment lambda job is detecting undeployed models from your provided model list and conduct auto-deployment in a schedule.