**Event-Driven S3 Data Ingestion With Node.js Lambda Function and Deploy it with Serverless**

Ingesting data upon the file creating on [S3](https://aws.amazon.com/s3/) bucket enables near real-time data ingestion. For example, you may need to ingest log files from applications or API monitoring tools as soon as they land on the bucket. Just to get it started, let’s move the file from the source bucket to the target as soon as it gets created in the source bucket by using the [node.js lambda function](https://docs.aws.amazon.com/lambda/latest/dg/programming-model.html).

We covered the same ingestion pattern with Python in the past. Here, we are using node.js for the lambda function and [serverless](https://serverless.com/framework/docs/providers/aws/guide/intro/) for deployment including the creation of source and target bucket.

**Prerequisites**

I won’t cover basic stuff because they are already covered in the past. If you need to brush up your knowledge, you can follow the link that was covered in my posts.

* Basic knowledge of S3 file download and upload with Node.js (see [Uploading and Downloading Files in S3 with Node.js](https://www.mydatahack.com/uploading-and-downloading-files-in-s3-with-node-js/)).
* Basic knowledge of how lambda function works and how to deploy it (see [Event-Driven Data Ingestion with AWS Lambda](https://www.mydatahack.com/event-driven-data-ingestion-in-aws-s3-to-s3/)).
* Basic knowledge of serverless framework (see [Advanced AWS Lambda Python Function Deployment with Serverless](https://www.mydatahack.com/advanced-aws-lambda-python-function-deployment-with-serverless/)).
* Basic knowledge of CloudFormation (example: [Creating Publicly Accessible RDS with CloudFormation](https://www.mydatahack.com/creating-publicly-accessible-rds-with-cloudformation/)).

**Event Payload**

By configuring the event notification on S3, it will send the event as the JSON format to the lambda function. The event in the lambda function argument looks like the Json object below (see further information [here](https://docs.aws.amazon.com/lambda/latest/dg/with-s3-example-upload-deployment-pkg.html)).

[cc lang="text" tab\_size="4" lines="-1" theme="mac-classic" line\_numbers="false"]

[/cc]

**Lambda Function**

This should be relatively straight forward. The source bucket and file names are retrieved from the payload object (for further information, see [Uploading and Downloading Files in S3 with Node.js](https://www.mydatahack.com/uploading-and-downloading-files-in-s3-with-node-js/)).

<pre class=" toolbar:2 lang:js decode:true" >

</pre>

**Lambda Function Deployment**

With Serverless, we can define function, events and resource creations with config files or parameters in the [serverless.yml](https://serverless.com/framework/docs/providers/aws/guide/serverless.yml/) file. It is a simplified version of CloudFormation template for deploying lambda function.

In the previous post, we only deployed the function with serverless.yml. This means we had to create source and target buckets and configure event on the source as well as creating lambda permissions for the bucket to invoke the function.

To create resources, you can use the raw CloudFormation template syntax in YAML under [the resources section](https://serverless.com/framework/docs/providers/aws/guide/resources/). Whatever you defined in serverless.yml gets converted into the CloudFormation template which can be found in the .serveless folder.

In this way, we can create all the appropriate resources and permissions when we deploy the function and update on any resource or permission setting can be changed when you update the serverless.yml file.

However, there is a catch to this. Because of the way CloudFormation creates resources, we cannot add the event notification to the source bucket upon creation. With the NotificationConfiguration property in the source bucket, you will get the error, ***unable to validate the following destination configuration***. You can read more about this issue [here](https://aws.amazon.com/premiumsupport/knowledge-center/unable-validate-circular-dependency-cloudformation/).

The easiest way to avoid this error is to deploy the function without NoficationConfiguration first. Once you deploy the lambda function first time, you can add it to the resource and deploy it again. The second deployment is seen as a stack update and it will add the event notification to the source bucket.

There will be a better way than deploying the function twice with update. But, I think it is the simplest workaround.

Here is the template. For the IAM roles statement, we separated it into a permissions.yml file in the config folder (see [here](https://www.mydatahack.com/advanced-aws-lambda-python-function-deployment-with-serverless/) for getting values from config files for serverless.yml).

**serverless.yml**

[cc lang="yaml" tab\_size="4" lines="-1" theme="GeSHi"]

[/cc]

**permission.yml**

[cc lang="yaml" tab\_size="4" lines="-1" theme="GeSHi"]

[/cc]