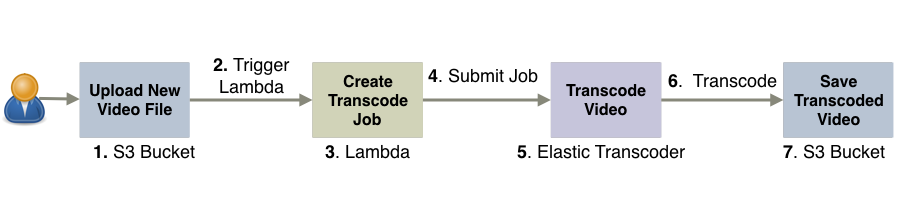
# Lesson 1

In lesson 1 we are going to create the engine of our YouTube clone. Make sure you can log in to the AWS console and follow the instructions given below.

**Note: please create all your resources in the N. Virginia Region (us-east-1)**

This is the system we will end up with at the end of this lesson



## Create 2 S3 buckets

Let’s begin by creating two buckets in S3. The first bucket will serve as the upload bucket for new videos. The second bucket will contain transcoded videos put there by the Elastic Transcoder.

* To create a bucket, in the AWS console click on S3, and then click Create Bucket (Figure 1).
* Type in a name for the bucket (e.g. serverless-video-upload), and choose a region (US Standard).
* Click Create to save your bucket.
* Repeat the process again to create another bucket (e.g. serverless-video-transcoded).

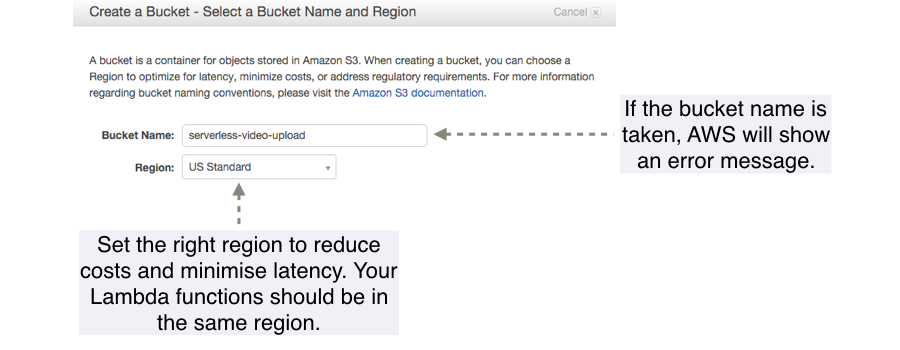


Figure : Create an S3 Bucket

## Create an IAM Role for your first Lambda function

Now we need to create an IAM role for our future Lambda functions. This role will allow functions to interact with S3 and the Elastic Transcoder.

* In the AWS console, click Identity & Access Management and then click Roles.
* Click Create New Role and name it lambda-s3-execution-role. Click Next Step to proceed to Role Type selection.
* Under the AWS Service Roles click AWS Lambda and then select the following two policies:
  + AWSLambdaExecute
  + AmazonElasticTranscoderJobsSubmitter
* Click Next Step to attach both policies to the role and then click Create Role to save.
* You will be taken back to the role summary page. Click lambda-s3-execution-role again to see the two attached policies (figure 2).

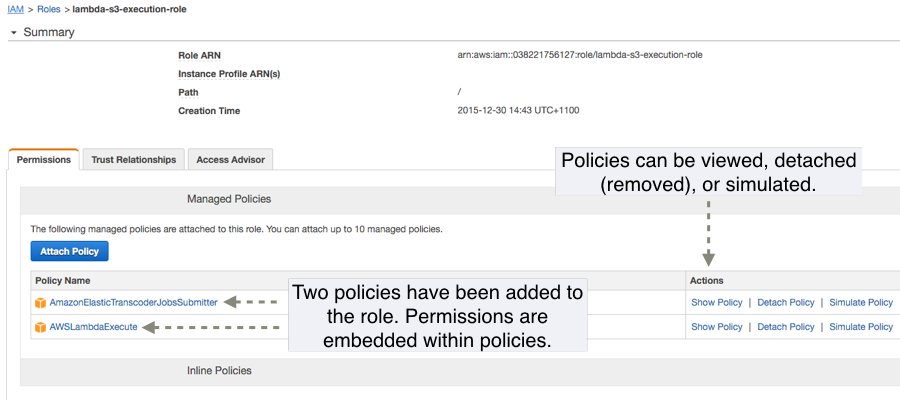


Figure : IAM Role

## Configure Elastic Transcoder

Now we need to set up an Elastic Transcoder pipeline to perform video transcoding to different formats and bitrates.

* In the AWS console click on Elastic Transcoder and then click Create a New Pipeline.
* Give your pipeline a name, such as 24 Hour Video, and specify the input bucket, which in our case is the first upload bucket.
* Leave the IAM role as it is. Elastic Transcoder creates a default IAM role automatically.
* Under Configuration for Amazon S3 Bucket for Transcoded Files and Playlists specify the transcoded videos bucket, which in our case is serverless-video-transcoded. The Storage Class can be set to Standard.
* We are not generating thumbnails but we should still select a bucket and a storage class. Use the second, transcoded videos bucket for it again (figure 4).
* Click Create Pipeline to save.

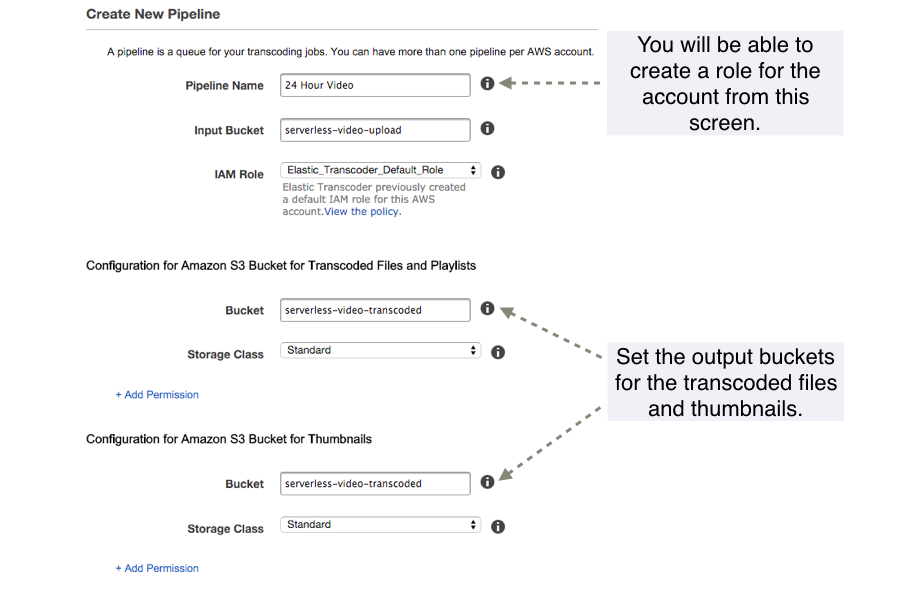


Figure : Elastic Transcoder Configuration

## Create Lambda function in AWS

It is finally time to create the first Lambda function although we are not going to provide an implementation for it just yet.

* In the AWS console, click Lambda, and then click Create a Lambda Function. Skip over the blueprint.
* Name the function transcode-video and make sure that Node.js 4.3 is selected in the *Runtime* dropdown.
* In the space for the function code enter two curly braces: {}. If you leave function code empty you will not be able to save.
* Under Role select lambda-s3-execution-role.
* Click Next to go the Review screen and from there click Create function to finish.

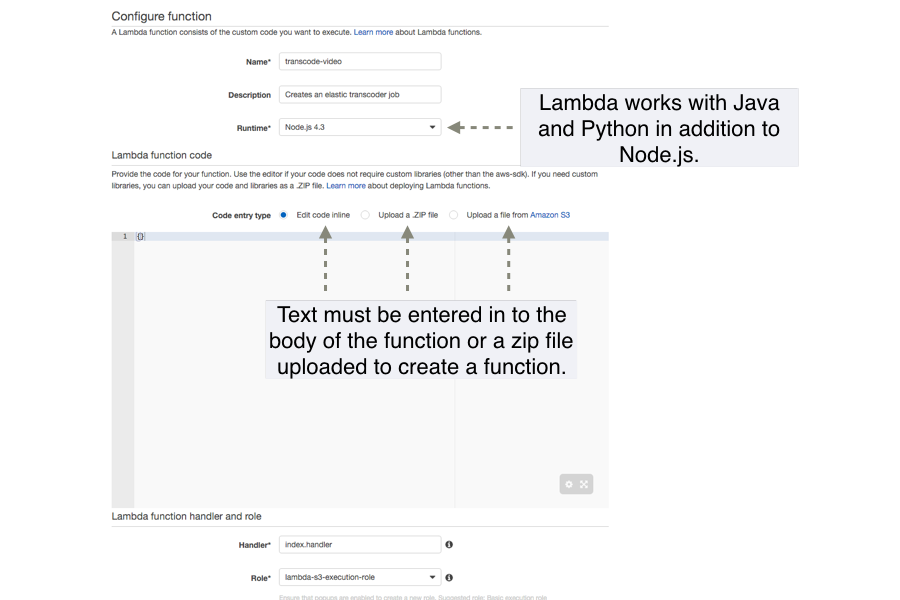


Figure 4: Create a Lambda function in AWS

## Prepare & Deploy Lambda

Finally, we can have a look at the actual Lambda function and deploy it to AWS.

* Open the Lambda function provided in lesson 1 in your favourite text editor.
* In the config.js file change **ELASTIC\_TRANSCODER\_PIPELINE\_ID** to correspond to your Elastic Transcoder pipeline ID (you can find it in the Elastic Transcoder console). See figure 5.

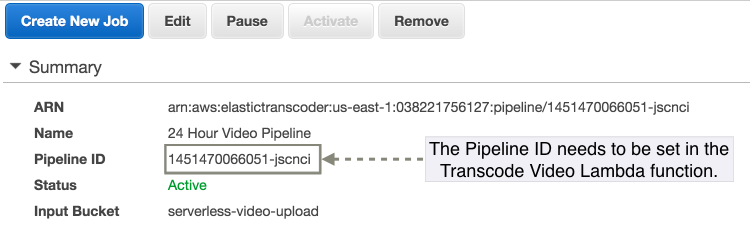


Figure : You need to get your Pipeline ID and add it to the function

* In the terminal, change to the directory of the function and type *npm run predeploy* to create a ZIP file of the function.
  + If you are on Windows, you will have to zip up all the files via the GUI or using a utility such as 7zip.
* In the AWS console click Lambda, select your function, and click Upload.

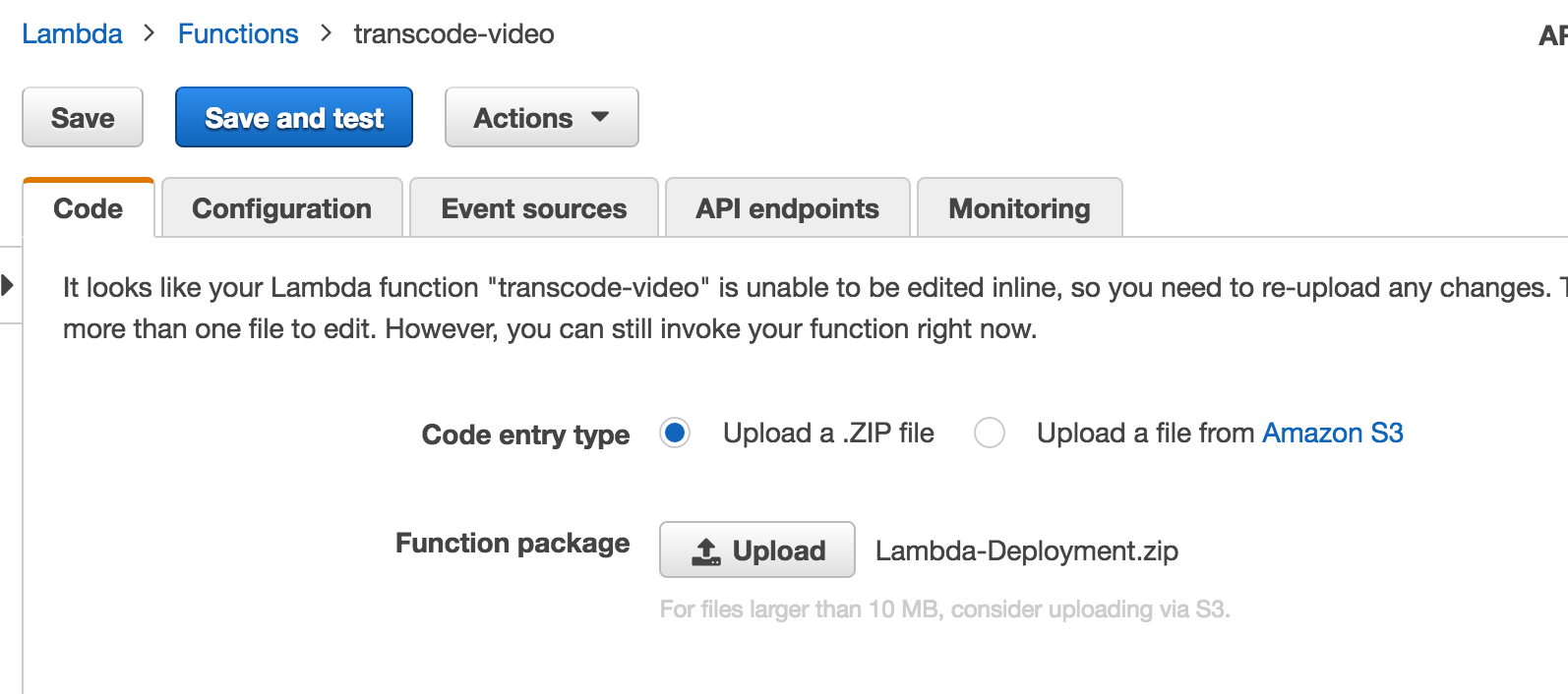


Figure : Uploading the function

* Select the ZIP of the Lambda function you had created earlier.
* Click the Save button to upload the function.

## Connect S3 to Lambda

The last step before we can test the function in AWS is to connect S3 to Lambda (figure 7).

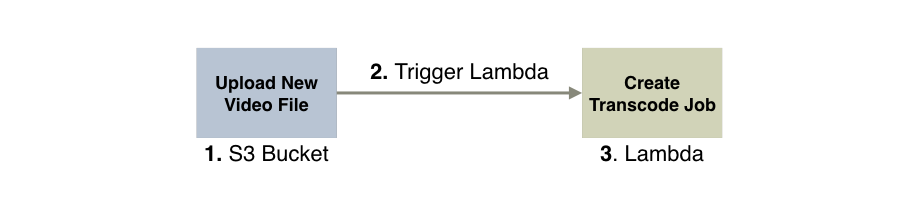


Figure : S3 will invoke our Lambda function

* Open the upload bucket (e.g. serverless-video-upload) in the AWS console, select Properties, expand Events, and click Add Notification.
* Give your event a name, such as Video Upload, and in the Events dropdown select ObjectCreated (All).
* Click the Lambda function radio button, right below it select the Lambda function from the dropdown and save (figure 8).

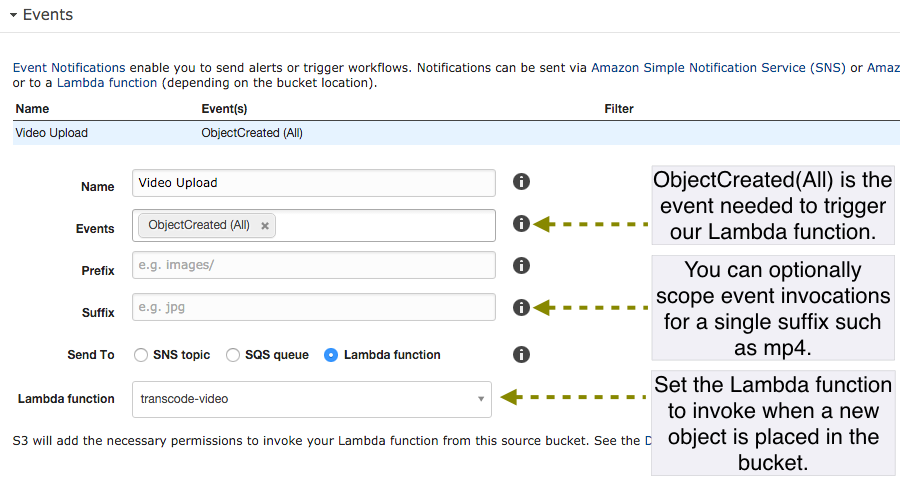


Figure : Configure S3 event

## Testing in AWS

To test the function in AWS, upload a video to the upload bucket. To do this click the bucket, click Actions and then select Upload (figure 9).

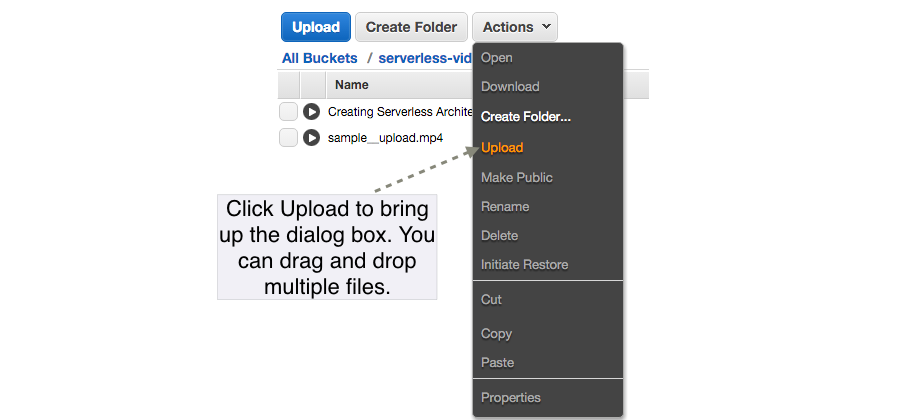


Figure : You can upload directly to a bucket

After a period of time, you should see three new videos appear in the transcoded videos bucket. These files should appear in a folder rather than in the root of the bucket (figure 10).

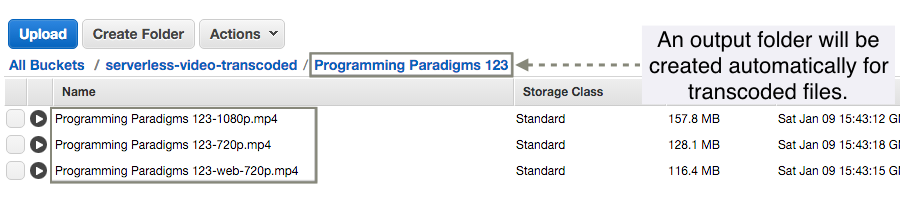


Figure : The files should appear in the bucket after a certain time