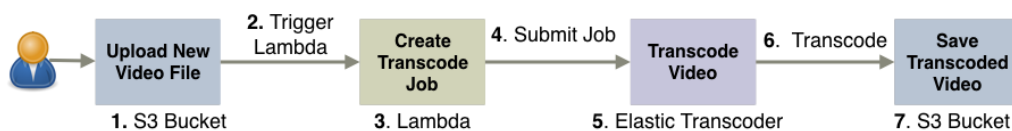


LESSON 1

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.

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



NOTE: PLEASE CREATE ALL YOUR RESOURCES IN THE N. VIRGINIA REGION (US-EAST-1)

1. SET YOUR REGION TO US. EAST (N. VIRGINIA)

Before we kick-off the build, log in to the AWS console, and set your region to US East (N. Virginia).

Please make sure that all resources & services you create are in the same region from here onwards.

2. 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**.
- Enter a **Bucket Name** (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).

Create bucket

1 Name and region 2 Set properties 3 Set permissions 4 Review

Name and region

Bucket name ⓘ

serverless-video-upload

Region

US East (N. Virginia)

Copy to an existing bucket

24 Buckets

Create Cancel Next

If the bucket name is taken, AWS will show an error message

Set the right region to reduce costs and minimise latency. Your Lambda functions should be in the same region.

3. MODIFY BUCKET POLICY

We need to make our transcoded videos publically accessible.

- In S3 click on the **second** bucket you have created (this will be the serverless-video-transcoded bucket).
- Expand **Permissions**
- Click **Add bucket policy**
- Type in the following to the bucket policy (you can copy below text form step3-bucket-policy.txt):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AddPerm",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::<YOUR-BUCKET-NAME>/*"
    }
  ]
}
```

Make sure to substitute YOUR-BUCKET-NAME with the actual name of your transcoded bucket.

- Click **Save**

Bucket policy editor ARN: arn:aws:s3:::

Type to add a new policy or edit an existing policy in the text area below.

Delete

Cancel

Save

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Sid": "AddPerm",
6       "Effect": "Allow",
7       "Principal": "*",
8       "Action": "s3:GetObject",
9       "Resource": "arn:aws:s3:::serverless-video-transcoded/*"
10    }
11  ]
12 }
13
```

4. 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 **Select Role Type** click **AWS Lambda**

Select Role Type

AWS Service Roles

- Amazon EC2**
Allows EC2 instances to call AWS services on your behalf. [Select](#)
- AWS Directory Service**
Allows AWS Directory Service to manage access for existing directory users and groups to AWS services. [Select](#)
- AWS Lambda**
Allows Lambda Function to call AWS services on your behalf. [Select](#)
- Amazon Redshift**
Allows Amazon Redshift Clusters to call AWS services on your behalf. [Select](#)
- Amazon API Gateway**
Allows API Gateway to call AWS resources on your behalf. [Select](#)

☐ **Role for Cross-Account Access**

☐ **Role for Identity Provider Access**

- 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:

Summary

Role ARN: arn:aws:iam::038221756127:role/lambda-s3-execution-role

Instance Profile ARN(s): /

Path: /

Creation Time: 2015-12-30 14:43 UTC+1100

Permissions | Trust Relationships | Access Advisor

Managed Policies

The following managed policies are attached to this role. You can attach up to 10 managed policies.

[Attach Policy](#)

Policy Name	Actions
AmazonElasticTranscoderJobsSubmitter	Show Policy Detach Policy Simulate Policy
AWSLambdaExecute	Show Policy Detach Policy Simulate Policy

Inline Policies

Policies can be viewed, detached (removed), or simulated.

Two policies have been added to the role. Permissions are embedded within policies.

5. 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. (*see figure below*)
- 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 was *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.
- Click **Create Pipeline** to save.

Create New Pipeline

A pipeline is a queue for your transcoding jobs. You can have more than one pipeline per AWS account.

Pipeline Name:

Input Bucket:

IAM Role:

Elastic Transcoder previously created a default IAM role for this AWS account. [View the policy.](#)

Configuration for Amazon S3 Bucket for Transcoded Files and Playlists

Bucket:

Storage Class:

[+ Add Permission](#)

Configuration for Amazon S3 Bucket for Thumbnails

Bucket:

Storage Class:

[+ Add Permission](#)

6. 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**.
- Click **Blank Function**.
- Skip over configure triggers page (click **Next**).
- **Name** the function **transcode-video** and make sure that **Node.js 4.3** is selected in the **Runtime** dropdown.
- Under Role select **lambda-s3-execution-role**.
- Set the **timeout** to 30 seconds.
- Click **Next** to go the Review screen and from there click **Create function** to finish.

7. PREPARE & DEPLOY LAMBDA

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

- **Install npm packages**

In the terminal / command-prompt, change to the directory of the function:

```
cd lab-1/lambda/video-transcoder
```

Install npm packages by typing:

```
npm install
```

- **Zip Lambda function**

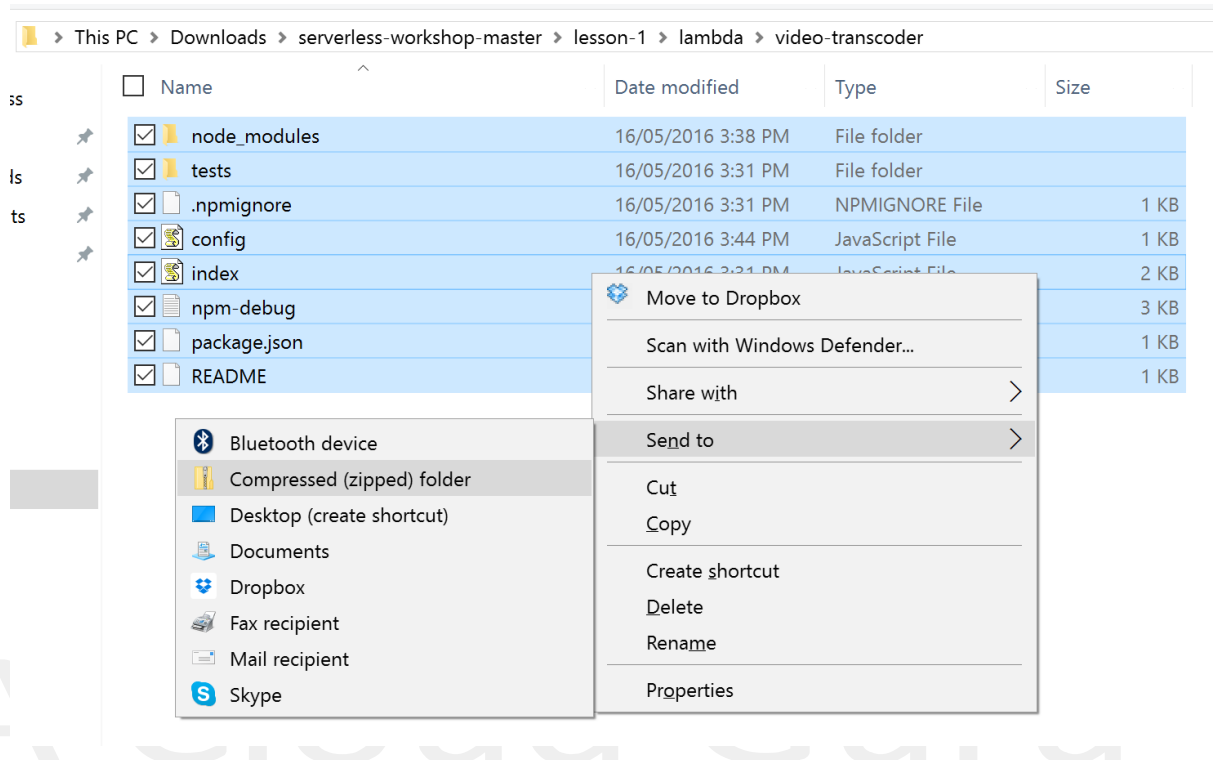
For OS X / Linux Users

Now create create a ZIP file of the function, by typing:

```
npm run predeploy
```

For Windows

You will need to **zip up all the files** in the **lab-1/lambda/video-transcoder** folder via the Windows Explorer GUI, or using a utility such as 7zip. (**Note: don't zip the video-transcoder folder. Zip up the files inside of it**).



- In the AWS console click **Lambda**
- Select your function
- Change the **code entry type** to **Upload a .ZIP file**
- Click Upload:

Code

Configuration

Triggers

Monitoring

Code entry type

Upload a .ZIP file

Function package

Upload

For files larger than 10 MB, consider uploading via S3.

- Select the ZIP file of the Lambda function you had created earlier.
- Scroll down to the environment variables section.
- Add an environment variable with Key **ELASTIC_TRANSCODER_REGION** and set its value to **us-east-1**
- Add another environment variable with Key **ELASTIC_TRANSCODER_PIPELINE_ID** and set the Value to correspond to your Elastic Transcoder pipeline ID (you can find it in the Elastic Transcoder console by clicking on the details icon):

Create New Pipeline Create New Job Edit Pause Activate Remove					
Filter: <input type="text"/>					
Viewing 1 item					
	Name	Input Bucket	Bucket for Transcoded Files	Bucket for Thumbnails	Status
<input type="checkbox"/>	24-hour-video	peter-upload-bucket	peter-transcoded-bucket	peter-transcoded-bucket	Active

This is the Pipeline ID you need to copy into the environment variable above.

Create New Job Edit Pause Activate Remove	
▼ Summary	
ARN	arn:aws:elastictranscoder:us-east-1:038221756127:pipeline/1451470066051-jscnci
Name	24 Hour Video Pipeline
Pipeline ID	1451470066051-jscnci
Status	Active
Input Bucket	serverless-video-upload

The Pipeline ID needs to be set in the Transcode Video Lambda function.

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

Your environment variables should look a bit like this but should have the pipeline id of your elastic transcoder pipeline.

You can define Environment Variables as key-value pairs that are accessible from your function code. These are useful to store configuration settings without the need to change function code. [Learn more](#). For storing sensitive information, we recommend encrypting values using KMS and the console's encryption helpers.

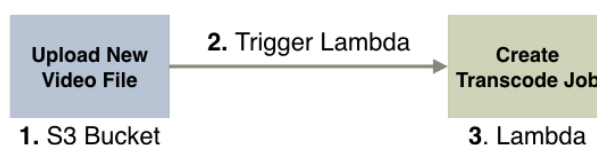
Enable encryption helpers ☐

Environment variables	
ELASTIC_TRANSCODER_PIPELINE_ID	1490833093197-vuzh4a
ELASTIC_TRANSCODER_REGION	us-east-1
Key	Value

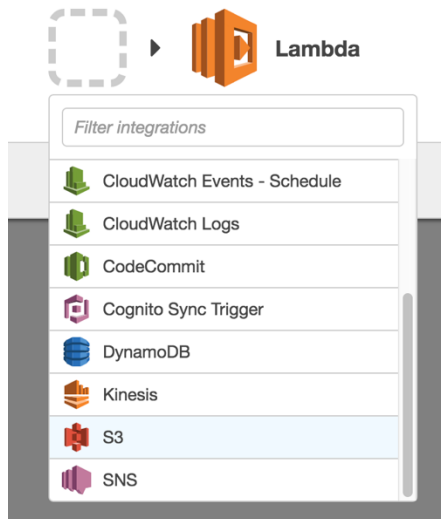
- Click the **Save** button to upload the function and set the environment variables.

8. CONNECT S3 TO LAMBDA

The last step before we can test the function in AWS is to connect S3 to Lambda. S3 will invoke our lambda function:





- In the Lambda console click on the **Triggers** tab and click **Add trigger**.
- In the popup dialog, click to the left of the orange lambda logo and scroll down the list until you find S3 and select that.



- Select the upload bucket (e.g. serverless-video-upload).
- In the event type dropdown, select **ObjectCreated (All)**.
- Press **Submit** and AWS will link your s3 bucket and lambda function.

Add trigger ✕

Configure your Lambda function **transcode-video** to respond to events from the selected trigger. Click on the box below to select your trigger type.

S3  ▶  Lambda

Bucket danparker-sydney-serve ⓘ

Event type Object Created (All) ⓘ

Prefix e.g. images/ ⓘ

Suffix e.g. .jpg ⓘ

Lambda will add the necessary permissions for Amazon S3 to invoke your Lambda function from this bucket. [Learn more](#) about the Lambda permissions model.

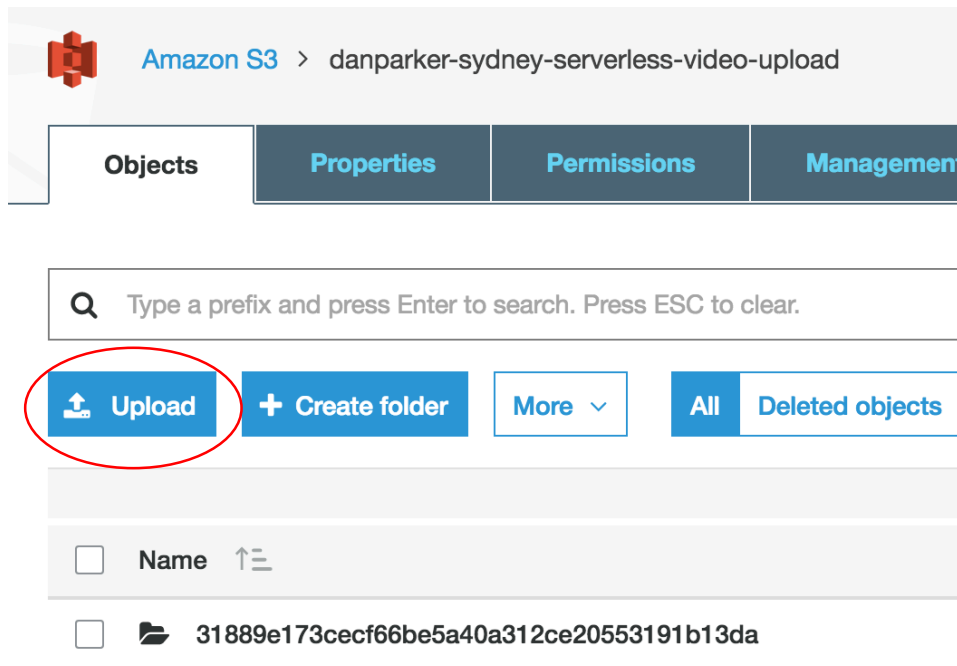
Enable trigger ☒ ⓘ

Cancel Submit

9. TESTING IN AWS

Need a video for testing? Download our test videos from: <http://bit.ly/1TEkvuR>

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**:



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:

Upload

Create folder




More

All

Deleted objects

US East (N. Virginia)

Viewing 1 to 3

<input type="checkbox"/>	Name	Last modified	Size	Storage class
<input type="checkbox"/>	 mogali-1080p.mp4	Mar 30, 2017 7:55:23 PM	4.2 MB	Standard
<input type="checkbox"/>	 mogali-720p.mp4	Mar 30, 2017 7:55:24 PM	1.9 MB	Standard
<input type="checkbox"/>	 mogali-web-720p.mp4	Mar 30, 2017 7:55:24 PM	1.9 MB	Standard

Congratulations – you now have your very own serverless video transcoding pipeline!

Get Your Hands Dirty

At the moment, *24-Hour Video* is functional but it has a number of limitations that have been left for you to solve as an exercise. See you if you can implement a solution for the following problems:

1. A file with more than one period in its name (for example, *Lecture 1.1 – Programming Paradigms.mp4*) is going to produce transcoded files with truncated names. Implement a fix it so that filenames with multiple periods work.
2. Currently, any file uploaded to the upload bucket will trigger the workflow. The Elastic Transcoder, however, will fail if it's given invalid input (for example, a file that is not a video). Modify the first Lambda function to check the extension of the uploaded file and only submit avi, mp4, or mov files to Elastic Transcoder. Any invalid files should be deleted from the bucket.
3. The files in the upload bucket are going to remain there until you delete them. Come up with a way to clean up the bucket automatically after 24 hours. You might want to have a look at the Lifecycle options in S3 for ideas.
4. The current system creates three transcoded videos that are very similar. The main difference between them is the resolution and bitrate. To make the system more varied, add support for HLS and webm formats.

A Cloud Guru