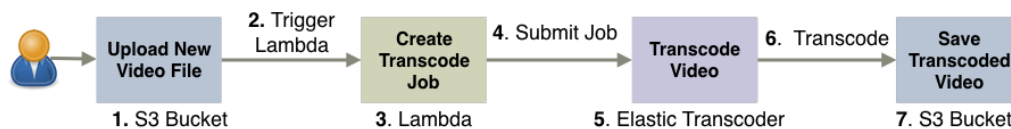


## LESSON 1

In lesson 1, we are going to create the engine of our YouTube clone. Make sure you can log into 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. VIRIGINA)

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 on.**

### 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 the **region: US East (N. Virginia)**.
- Click **Create** to save your bucket.
- Repeat the process again to create another bucket (e.g. *serverless-video-transcoded*).
- Make a note of the bucket names, as you will be using them throughout this workshop.

The screenshot shows the 'Create bucket' wizard in the AWS Management Console. It has four steps: 1. Name and region, 2. Set properties, 3. Set permissions, and 4. Review. In the 'Name and region' step, the 'Bucket name' field contains 'serverless-video-upload' and the 'Region' dropdown is set to 'US East (N. Virginia)'. Below these fields, there is a checkbox for 'Copy from an existing bucket' and a dropdown menu showing '24 Buckets'. At the bottom are 'Create', 'Cancel', and 'Next' buttons. Two callout boxes are present: one pointing to the region dropdown with the text 'Set the right region to reduce costs and minimize latency. Your Lambda functions should be in the same region.', and another pointing to the bucket name field with the text 'If the bucket name is taken, AWS will show an error message'.

### 3. MODIFY BUCKET POLICY

We need to make our transcoded videos publicly accessible.

- In S3 click on the **second** bucket you have created (this will be the serverless-video-transcoded bucket).
- Click on the **Permissions** tab
- Click **Bucket Policy**
- Enter the following to the bucket policy (you can copy below text from 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 serverless-video-transcoded bucket.**

- Click **Save**

Bucket policy editor ARN: arn:aws:s3::: [redacted]  
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's **Services** tab, click **IAM** under **Security, Identity & Compliance**, and then click **Roles** from the left navigation menu.
- Click **Create Role**
- In the **Trust** step, choose **AWS Service** and **Lambda**, and then click **Next: Permissions**

##### Create role

1

Trust

2

Permissions

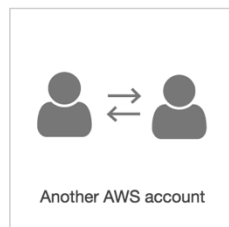
3

Review

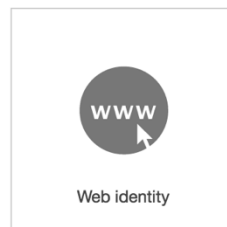
##### Select role type



AWS service



Another AWS account



Web identity



Saml 2.0 federation

Allows AWS services to perform actions on your behalf. [Learn more](#)

Choose the service that will use this role

<a href="#">API Gateway</a>	<a href="#">Data Pipeline</a>	<a href="#">IoT</a>	<a href="#">Service Catalog</a>
<a href="#">Auto Scaling</a>	<a href="#">Directory Service</a>	<b><a href="#">Lambda</a></b>	
<a href="#">Batch</a>	<a href="#">DynamoDB</a>	<a href="#">Lex</a>	
<a href="#">CloudFormation</a>	<a href="#">EC2</a>	<a href="#">Machine Learning</a>	
<a href="#">CloudHSM</a>	<a href="#">EC2 Container Service</a>	<a href="#">OpsWorks</a>	
<a href="#">CloudWatch Events</a>	<a href="#">EMR</a>	<a href="#">RDS</a>	
<a href="#">CodeBuild</a>	<a href="#">Elastic Beanstalk</a>	<a href="#">Redshift</a>	
<a href="#">CodeDeploy</a>	<a href="#">Elastic Transcoder</a>	<a href="#">SMS</a>	
<a href="#">Config</a>	<a href="#">Glue</a>	<a href="#">SNS</a>	
<a href="#">DMS</a>	<a href="#">Greengrass</a>	<a href="#">SWF</a>	

\* Required

Cancel

Next: Permissions

- In the Permissions step, search for and check the boxes next to:
  - **AWSLambdaExecute**
  - **AmazonElasticTranscoderJobsSubmitter**

- **Note: Make sure the names you select match exactly what is shown here.**
- Click **Next: Review** to attach both policies to the role.
- In the Review step, name the role **lambda-s3-execution-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:

Roles > lambda-s3-execution-role

## Summary Delete role

Role ARN: arn:aws:iam::841397984957:role/lambda-s3-execution-role

Role description: Edit

Instance Profile ARNs

Path: /

Creation time: 2017-09-27 19:41 CDT

Permissions | Trust relationships | Access Advisor | Revoke sessions

Attach policy Attached policies: 2

Policy name	Policy type	
AmazonElasticTranscoderJobsSubmitter	AWS managed policy	✕
AWSLambdaExecute	AWS managed policy	✕

## 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's **Services** tab, click on **Elastic Transcoder** under **Media Services**, 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 bucket, (e.g. *serverless-video-upload*).
- 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*.
- Set the **Storage Class** to **Standard**.
- We are not generating thumbnails but we should still select a bucket and a storage class. Use the second bucket, (*serverless-video-transcoded*) again, and once again set the **Storage Class** to **Standard**.
- Click **Create Pipeline** to save.
- Make note of the **Pipeline ID**. You'll need it soon.

**Create New Pipeline**

A pipeline is a queue for your transcoding jobs. You can have more than one pipeline per AWS account. You can use multiple pipelines to organize your transcoding jobs.

Pipeline Name:

Input Bucket:

IAM Role:

Elastic Transcoder creates a reusable, default IAM role. [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](#)

---

[▶ Notifications \(Optional\)](#)

[▶ Encryption \(Optional\)](#)

## 6. CREATE LAMBDA FUNCTION

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's **Services** tab, click **Lambda** under **Compute**, and then click **Create function**.
- Click **Author from scratch**.
- On the **Basic information** page, **Name** the function *transcode-video*.
- Under **Role**, select **Choose an existing role** and then **lambda-s3-execution-role**.
- Click **Create function**.
- Once the function is created, in the **Basic settings** section, set the **Timeout** to 0 minutes, 30 seconds.
- At the top of the page, click **Save**.

## 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**

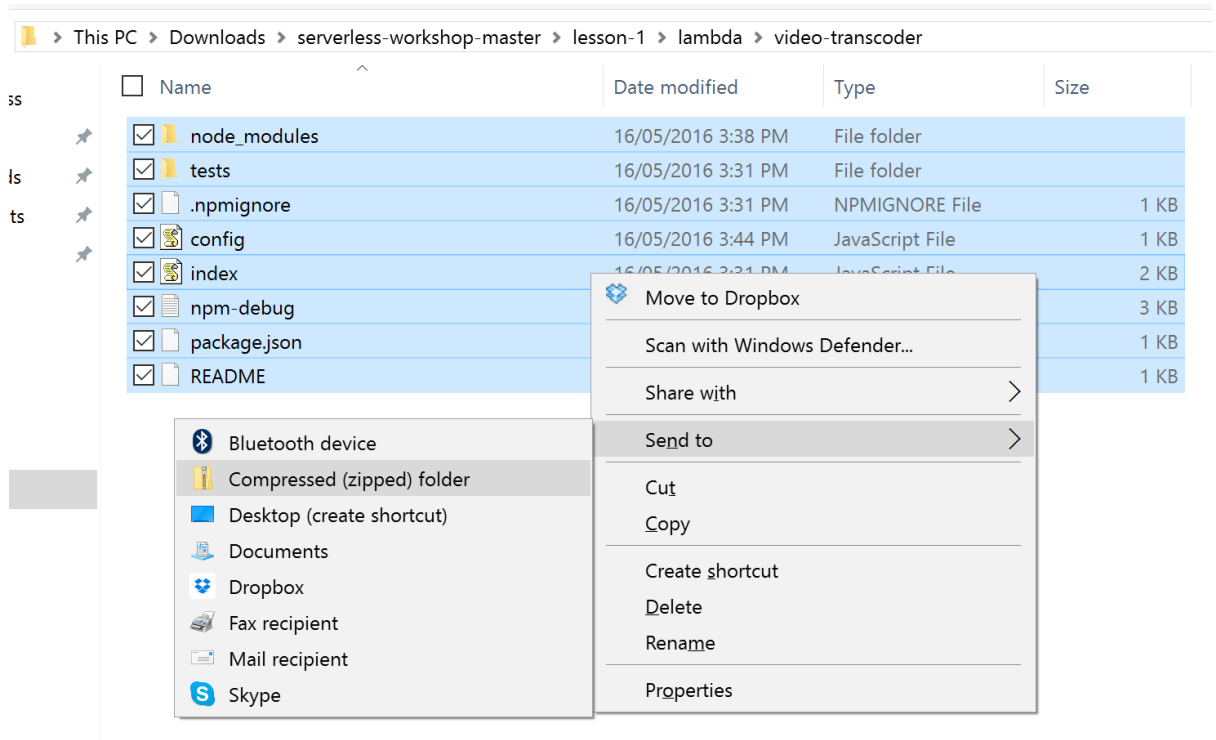
For OS X / Linux Users

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

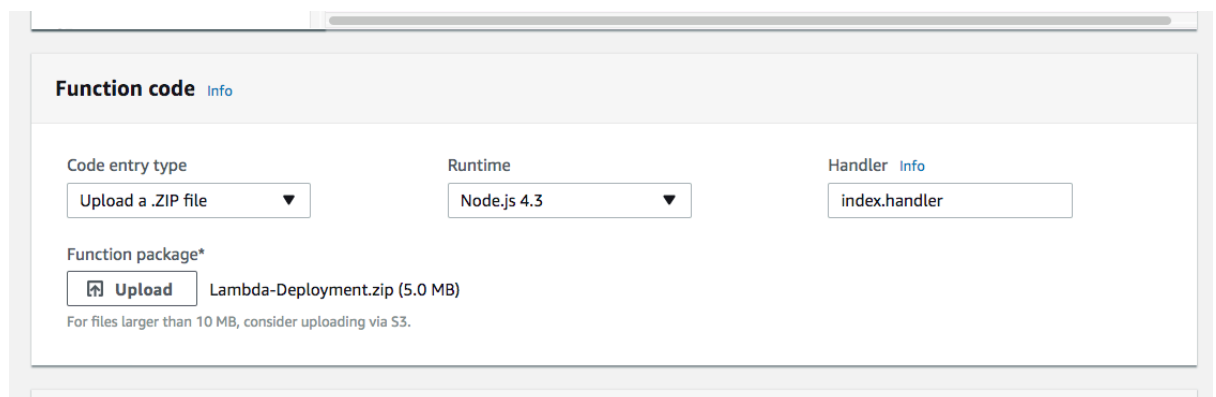
```
npm run predeploy
```

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**).



- Back in the AWS control panel, in the configuration for the *transcode-video* Lambda function:
- Under the **Function code** section, change the **Runtime** to **Node.js 4.3**.
- Set the **Code entry type** to **Upload a .ZIP file**.
- Click **Upload**:



- Select the .ZIP file of the Lambda function you created earlier.
- Scroll down to the **Environment variables**.
  - Add an environment variable with Key **ELASTIC\_TRANSCODER\_REGION** and set its **Value** to **us-east-1 (must be lower case)**
  - Add another environment variable with Key **ELASTIC\_TRANSCODER\_PIPELINE\_ID** and set its **Value** to be to your Elastic Transcoder pipeline ID from step 5. (you can find it in the Elastic Transcoder console by clicking on the details icon):

<a href="#">Create New Pipeline</a> <a href="#">Create New Job</a> <a href="#">Edit</a> <a href="#">Pause</a> <a href="#">Activate</a> <a href="#">Remove</a>					
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.

<a href="#">Create New Job</a> <a href="#">Edit</a> <a href="#">Pause</a> <a href="#">Activate</a> <a href="#">Remove</a>	
▼ 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 the pipeline id of your elastic transcoder pipeline will be different from the one shown.

▼ Environment variables

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.](#)

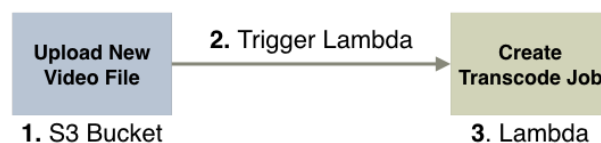
ELASTIC_TRANSCODER_PIPELINE_ID	1506564283288-6e22rz	<a href="#">Remove</a>
ELASTIC_TRANSCODER_REGION	us-east-1	<a href="#">Remove</a>
Key	Value	<a href="#">Remove</a>

► Encryption configuration

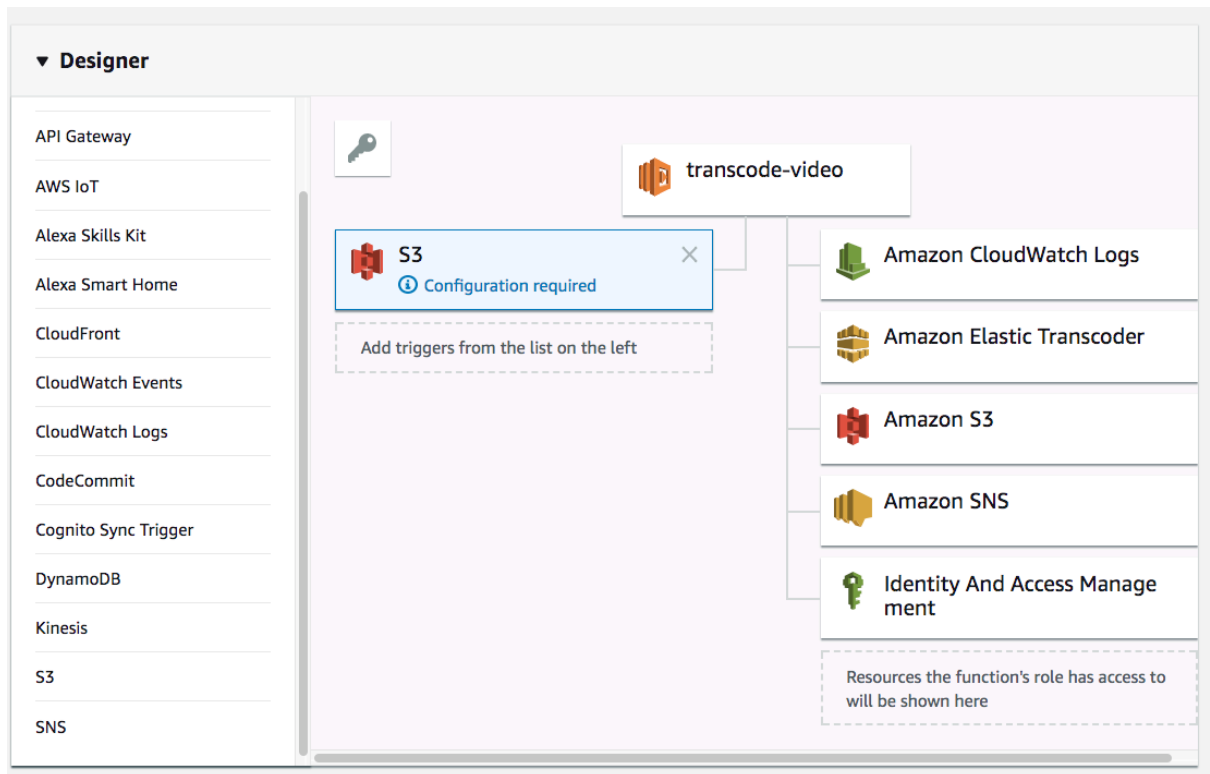
- Click the **Save** button at the top of the page to upload the function and set the environment variables.

## 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 when a new video is uploaded:



- On the same page, scroll up to the **Designer** section
- Click on **S3** in the **Add triggers** list on the left



- Scroll down to the **Configure triggers** section
- Select the upload bucket (e.g. *serverless-video-upload*).
- In the event type dropdown, select **Object Created (All)**.
- Press **Add**
- Now click on the **Save** button up the top and AWS will link your s3 bucket and lambda function.

### Configure triggers

**Bucket**  
Please select the S3 bucket that serves as the event source. The bucket must be in the same region as the function.

acg-sfb-upload-bucket ▼

**Event type**  
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.

Object Created (All) ▼

**Prefix**  
Enter an optional prefix to limit the notifications to objects with keys that start with matching characters.

e.g. images/

**Suffix**  
Enter an optional suffix to limit the notifications to objects with keys that end with matching characters.

e.g. .jpg

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

**Enable trigger**  
Enable the trigger now, or create it in a disabled state for testing (recommended).

☒

Cancel Add

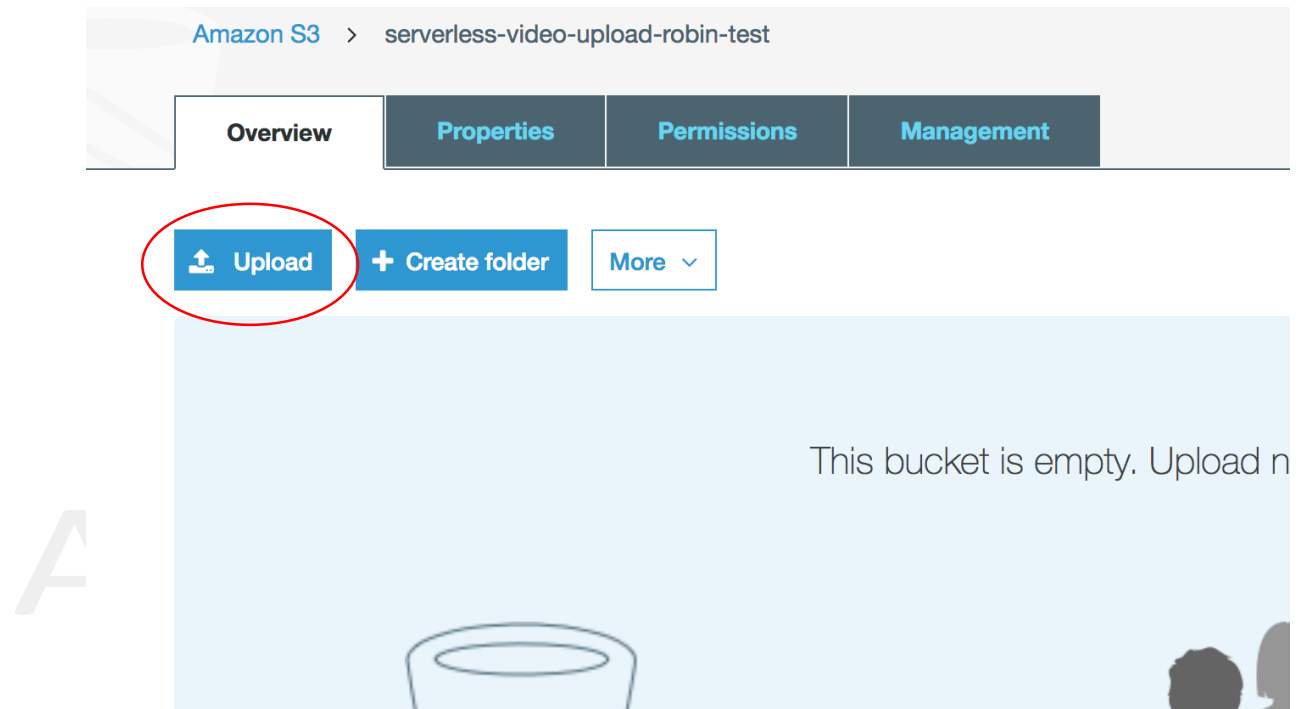


## 9. TESTING IN AWS

To test the function in AWS, upload a video to the upload bucket.

In the root directory of the serverless-workshop, there's a sample-videos.zip containing videos you can use to test the transcoder.

To do this, go to **S3**, navigate to the *upload* bucket, and then select **Upload**:



- Click **Add Files**, select a video file (an .avi, .mp4, or .mov), and click **Upload**. The file you selected should appear in the *upload* bucket.
- Navigate to the *transcoded* bucket, and after a short period of time (long enough to grab a cup of coffee, not long enough for a proper nap), you should see three new videos. These files will appear in a folder rather than in the root of the bucket:

A screenshot of the Amazon S3 console showing the 'transcoded' bucket. At the top, there are buttons for 'Upload', '+ Create folder', 'More', 'All', and 'Deleted objects'. The region is 'US East (N. Virginia)'. Below the buttons is a table with the following data:

	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!**

**Important:** Make sure that the files appear in the transcoded bucket before moving on to the next lesson. If they don't appear after a few minutes, double check each of the steps above.

## 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