

- **Project 2.2: Creating a Cloud-Front CDN Distribution to distribute a set of image files.**

Objective:

- This project aims at creating a Cloud-Front CDN Distribution to distribute a set of image files.

Used AWS Resources:

- Cloud-front Content distributed network(CDN).
- Simple Storage Service(S3).

End Result:

- Getting an image file through the Content delivery network.

Create bucket

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

Name and region

Bucket name ⓘ

test0011

Region

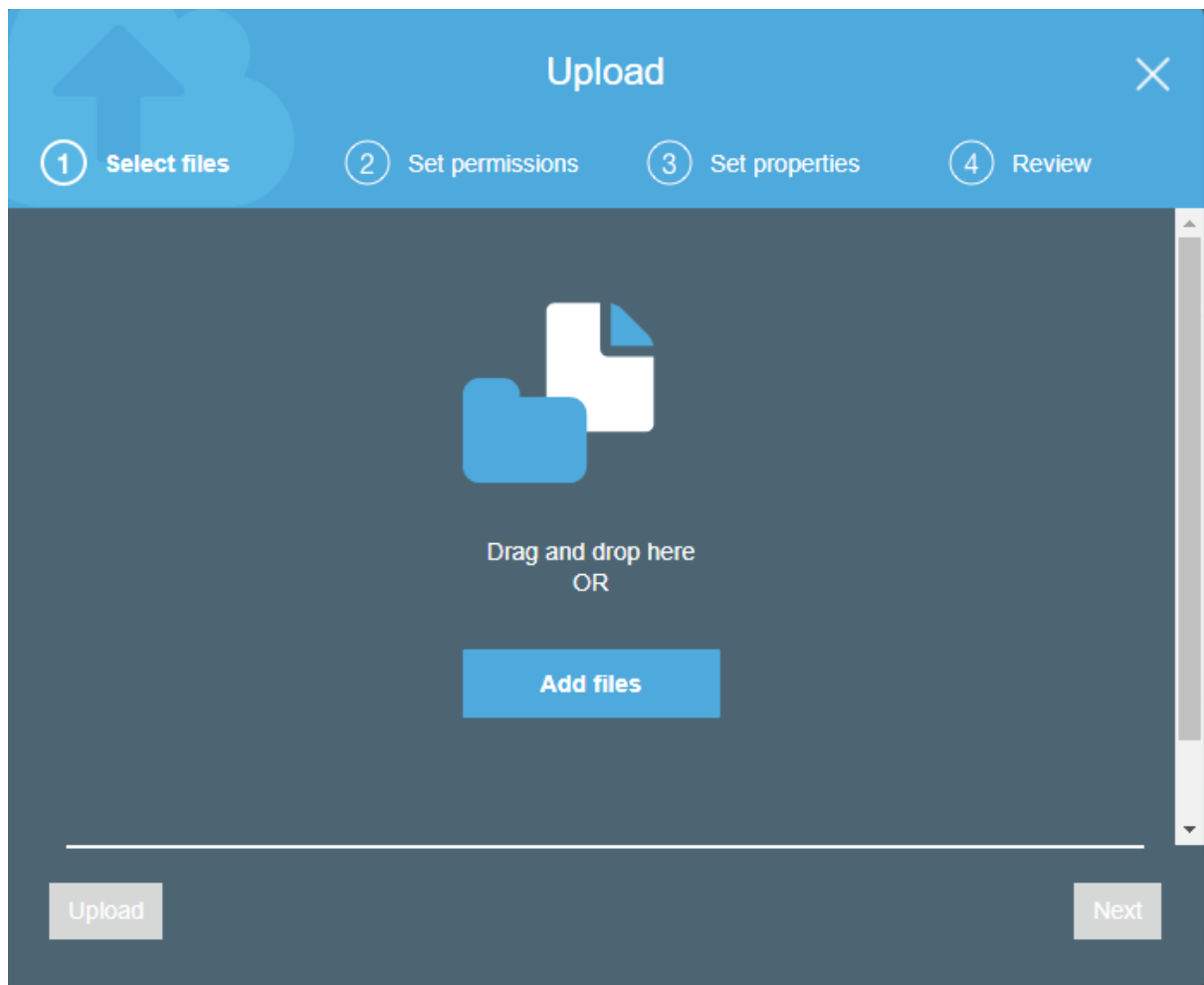
US East (N. Virginia) ▾

Copy settings from an existing bucket

You have no buckets 0 Buckets ▾

Create Cancel Next

- First create a bucket with a unique name.
- Select the specific region where you want to locate bucket.
- If you have any previous buckets, then you can copy and apply same settings to the creating bucket.
- Click on to the create button to create a bucket.
- If you want to set the permission of bucket, then click on the next button which is located at right bottom as shown in the above screenshot.



- After the creation of the bucket, upload required html file(testing.html) and image file(nature.jpg).
- To add files into bucket, click on the add files button then select files which you want to store into bucket.
- Then click on the upload button to upload selected files.

nature.jpg Latest version ▾

Overview Properties Permissions

Open Download Download as **Make public** Copy path

Owner
srinivasvl.vayalapalli99

Last modified
Sep 24, 2017 5:42:14 PM

Etag
b29259d837d4aaeef4b33c9dbc964a5b

Storage class
Standard

Server side encryption
None

Size
5081543

Link
<https://s3.amazonaws.com/test0011/nature.jpg>

- To make public the files, select the files and click on the make public button.
- <https://s3.amazonaws.com/test0011/nature.jpg> is the link to access the nature.jpg file from s3.
- Now we will create the cloud front content distribution to access the same image file which is stored in s3.
- For that we have to create one CDN web distribution, after that we will setup some of the settings over the cloud front.

Amazon CloudFront Getting Started

Either your search returned no results, or you do not have any distributions. Click the button below to create a new CloudFront distribution. network of edge locations that provide low latency and high data transfer speeds ([learn more](#))

Create Distribution

- if you aren't have any cloud front distributions, then the dash board of cloud front will looks like this as shown above screenshot.

Select a delivery method for your content.

Web

Create a web distribution if you want to:

- Speed up distribution of static and dynamic content, for example, .html, .css, .php, and graphics files.
- Distribute media files using HTTP or HTTPS.
- Add, update, or delete objects, and submit data from web forms.
- Use live streaming to stream an event in real time.

You store your files in an origin - either an Amazon S3 bucket or a web server. After you create the distribution, you can add more origins to the distribution.

[Get Started](#)

RTMP

Create an RTMP distribution to speed up distribution of your streaming media files using Adobe Flash Media Server's RTMP protocol. An RTMP distribution allows an end user to start playing a media file before the file has finished downloading from a CloudFront edge location. Note the following:

- To create an RTMP distribution, you must store the media files in an Amazon S3 bucket.
- To use CloudFront live streaming, create a web distribution.

[Get Started](#)

- There are two delivery methods available, those are web and RTMP.

Web:

- Web is normally used to speed up distribution of static and dynamic content like any. Html, CSS, php, and any graphics files.
- This is using https and http protocols.

RTMP:

- RTMP is normally used to speed up distribution of your streaming media server's RTMP protocol.
- We can get the live streaming of videos.

Create Distribution

Origin Settings

Origin Domain Name



Origin Path

— Amazon S3 Buckets —
test0011.s3.amazonaws.com ✓
— Elastic Load Balancers —
No Origins Available



Origin ID



Origin Custom Headers

Header Name

Value



Default Cache Behavior Settings

Path Pattern

Default (*)



Viewer Protocol Policy

- ☒ HTTP and HTTPS
☐ Redirect HTTP to HTTPS
☐ HTTPS Only



Allowed HTTP Methods

- ☒ GET, HEAD
☐ GET, HEAD, OPTIONS
☐ GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE



Cached HTTP Methods

GET, HEAD (Cached by default)



Cache Based on Selected
Request Headers

None (Improves Caching) ▼



- Creation of distribution network we have to set some of the origin settings.
- In origin domain name, we have to select s3 bucket name or some other elastic load balancer (ELB).
- Origin path could be the path of the folder that you had created in your s3 bucket. But this one is optional.
- Origin id could be taken a default name. if we want set any name, then we can do that.
- As of now no need to change any default cache behavior settings.

Create Distribution

Origin Settings

Origin Domain Name	<input type="text" value="test0011.s3.amazonaws.com"/>	
Origin Path	<input type="text" value="/test0011/nature"/>	Optional. If you want CloudFront to request your content directory name to the value of Origin Domain Name
Origin ID	<input type="text" value="S3-test0011/test0011/nature"/>	
Restrict Bucket Access	<input checked="" type="radio"/> Yes <input type="radio"/> No	
Origin Access Identity	<input checked="" type="radio"/> Create a New Identity <input type="radio"/> Use an Existing Identity	
Comment	<input type="text" value="access-identity-test0011.s3.amazonaws.com"/>	
Grant Read Permissions on Bucket	<input checked="" type="radio"/> Yes, Update Bucket Policy <input type="radio"/> No, I Will Update Permissions	
Origin Custom Headers	Header Name <input type="text"/>	Value <input type="text"/>

Default Cache Behavior Settings

Path Pattern	Default (*)	
Viewer Protocol Policy	<input checked="" type="radio"/> HTTP and HTTPS <input type="radio"/> Redirect HTTP to HTTPS <input type="radio"/> HTTPS Only	
Allowed HTTP Methods	<input checked="" type="radio"/> GET, HEAD <input type="radio"/> GET, HEAD, OPTIONS <input type="radio"/> GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE	
Cached HTTP Methods	GET, HEAD (Cached by default)	
Cache Based on Selected Request Headers	<input type="button" value="None (Improves Caching)"/>	

[Learn More](#)

CloudFront Distributions

[Create Distribution](#) [Distribution Settings](#) [Delete](#) [Enable](#) [Disable](#)

Viewing: Any Delivery Method Any State « < Viewing 1 to 1 of 1 Items > »

	Delivery Method	ID	Domain Name	Comment	Origin	CNAMEs	Status	State	Last Modified
	Web	E380P1Q1733Y4D	d2ufyi9bodem67v.cloudfront.net	-	test0011.s3.ar	-	In Progress	Enabled	2017-09-24 17:59 UTC

- These are the filled details in the distribution setting block.
- Here we can see the created web distributed cloud front.

AWS Policy Generator

The AWS Policy Generator is a tool that enables you to create policies that control access to [Amazon Web Services \(AWS\)](#) products and resources. For more information about creating policies, see [key concepts in Using AWS Identity and Access Management](#). Here are [sample policies](#).

Step 1: Select Policy Type

A Policy is a container for permissions. The different types of policies you can create are an [IAM Policy](#), an [S3 Bucket Policy](#), an [SNS Topic Policy](#), a [VPC Endpoint Policy](#), and an [SQS Queue Policy](#).

Select Type of Policy S3 Bucket Policy ▼

Step 2: Add Statement(s)

A statement is the formal description of a single permission. See [a description of elements](#) that you can use in statements.

Effect ☒ Allow ☐ Deny

Principal

Use a comma to separate multiple values.

AWS Service Amazon S3 ▼ ☐ All Services (*)

Use multiple statements to add permissions for more than one service.

Actions -- Select Actions -- ☒ All Actions (*)

Amazon Resource Name (ARN)

ARN should follow the following format: `arn:aws:s3:::<bucket_name>/<key_name>`.
Use a comma to separate multiple values.

[Add Conditions \(Optional\)](#)

Add Statement

- we have to generate a bucket policy so that we can provide particular permissions at bucket level.

You added the following statements. Click the button below to Generate a policy.

Principal(s)	Effect	Action	Resource	Conditions
• *	Allow	s3:*	arn:aws:s3:::test0011/*	None

Step 3: Generate Policy

A *policy* is a document (written in the [Access Policy Language](#)) that acts as a container for one or more statements.

Generate Policy

[Start Over](#)

- Click on the Generate button to create a new policy for the bucket.

Overview
Properties
Permissions
Management

Upload
Create folder
More

US East (N. Virginia)

Viewing 1 to 2

<input type="checkbox"/>	Name	Last modified	Size	Storage class
<input type="checkbox"/>	nature.jpg	Sep 24, 2017 10:58:07 PM	4.8 MB	Standard
<input type="checkbox"/>	testing.html	Sep 24, 2017 10:59:49 PM	202.0 B	Standard

Viewing 1 to 2

- We uploaded two files into the bucket named test0011.
- Those are the nature.jpg, testing.html

CloudFront Distributions > EUPNV11FTVI00

General

Origins

Behaviors

Error Pages

Restrictions

Invalidations

Tags

Edit

Distribution ID

ARN

Log Prefix

Delivery Method

Cookie Logging

Distribution Status

Comment

Price Class

AWS WAF Web ACL

State

Alternate Domain Names (CNAMEs)

SSL Certificate

Domain Name

Custom SSL Client Support

Supported HTTP Versions

IPv6

Default Root Object

Last Modified

Log Bucket

EUPNV11FTVI00

arn:aws:cloudfront::514602770466:distribution/EUPNV11FTVI00

-

Web

Off

Deployed

-

Use All Edge Locations (Best Performance)

-

Enabled

-

Default CloudFront Certificate (*.cloudfront.net)

dlyd67tq3ym6v.cloudfront.net

-

HTTP/2, HTTP/1.1, HTTP/1.0

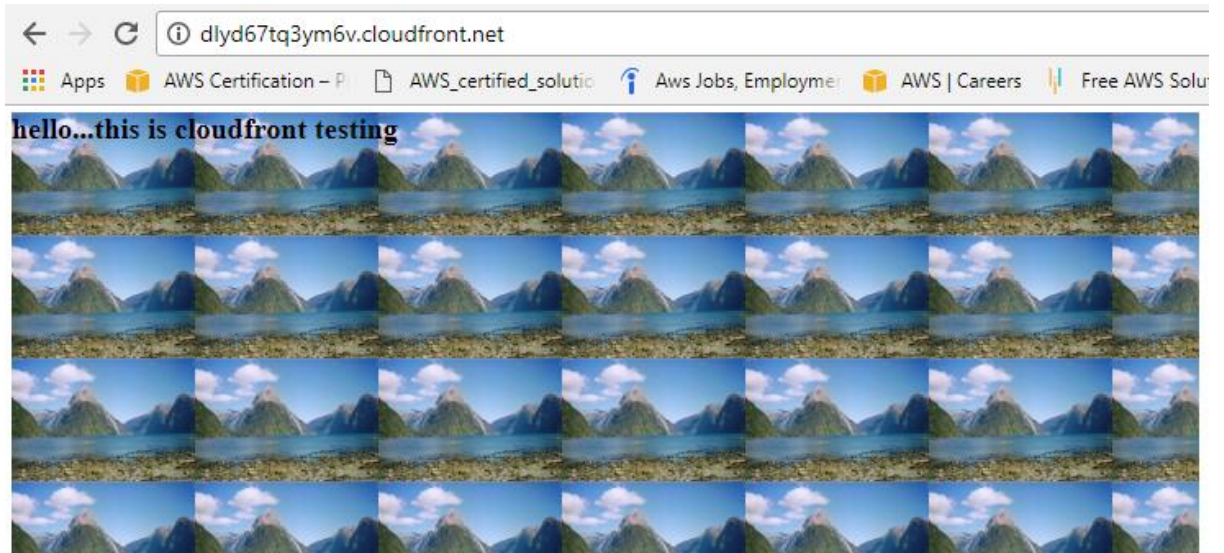
Enabled

testing.html

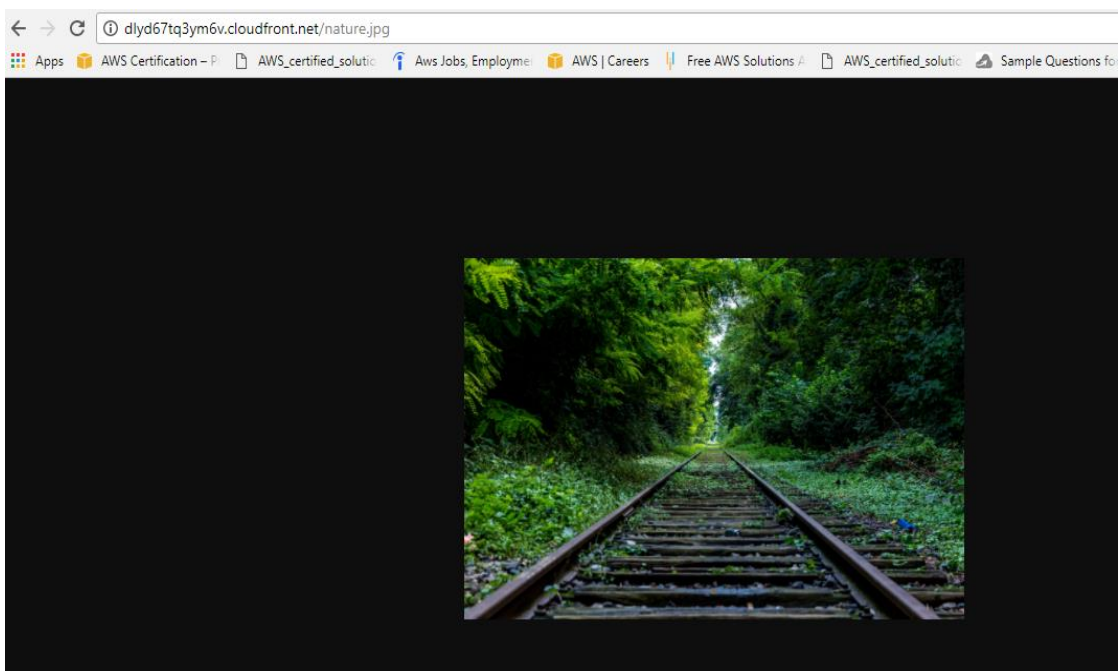
2017-09-24 22:59 UTC+5:30

-

- If we want to change any settings regarding the origin or change the bucket or add the elastic load balancer end point URL, those things we can do from here.



- After the deployment completion, we can simply copy that particular domain name and put it over web browser. Then we can get the output.



- This is the nature.jpg (image file) which we were uploaded in s3 bucket and accessing from the cloud-front without enter into the simple storage service(S3).
