



Lambda@edge and CloudFront Distribution

Lambda@Edge is an extension of AWS Lambda, designed to allow developers to run code closer to the end users of their applications, improving performance and reducing latency. It integrates with Amazon CloudFront, which is AWS's content delivery network (CDN), enabling you to execute code in AWS data centers located globally, called edge locations.

Key Features of Lambda@Edge:

- **Low Latency:** By executing code at CloudFront edge locations, it reduces the time it takes for a request to travel back to the origin server, enhancing response times for users around the world.
- **Event-Driven:** Lambda@Edge is triggered by specific CloudFront events, such as viewer requests, origin requests, viewer responses, and origin responses.
- **Serverless:** As with AWS Lambda, you don't need to manage servers. AWS takes care of scaling and infrastructure management.
- **Scalability:** Automatically scales based on the number of requests, handling any traffic load.
- **Security:** You can use Lambda@Edge to inspect and modify headers, authenticate users, and enforce authorization logic right at the edge.

Common Use Cases:

- **A/B Testing:** Serve different versions of your website to different users without deploying multiple versions of your site.
- **User Authentication and Authorization:** Authenticate users at the edge before requests reach your origin server.
- **Dynamic Content Manipulation:** Modify HTTP headers or generate HTTP responses dynamically based on request parameters.
- **SEO Optimization:** Modify URLs and redirect traffic based on user location or device type to optimize search engine rankings.

Lambda@Edge is particularly useful for applications that require quick, dynamic responses and need to be globally distributed to users.

A **CloudFront distribution** is a configuration setup within Amazon CloudFront, AWS's content delivery network (CDN) service. It defines how CloudFront should deliver your content (like web pages, images, videos, and other assets) to users across the globe.

Key Components of a CloudFront Distribution:

- **Origin:** The source of the content that CloudFront will distribute. This can be an Amazon S3 bucket, an HTTP server (like an EC2 instance), or an external web server.
- **Edge Locations:** These are data centers located around the world where CloudFront caches copies of your content to deliver it to users with low latency.
- **Cache Behavior:** Defines how CloudFront handles requests for content. You can set up multiple cache behaviors for different parts of your site or app, specifying the origin, caching policy, and response headers for each.
- **Distribution Types:**
 - **Web Distribution:** Used for serving static and dynamic content over HTTP or HTTPS, like websites, APIs, or media files.
 - **RTMP Distribution:** (Deprecated) Used for streaming media files using Adobe Flash Media Server's RTMP protocol.
- **Domain Name:** CloudFront assigns a unique domain name to your distribution, which you can use in your URLs or configure your custom domain to point to it.
- **SSL/TLS Configuration:** You can secure your content by using HTTPS and integrating your own SSL/TLS certificates.
- **Access Control:** CloudFront offers various security features, like signed URLs, signed cookies, and geoblocking, to restrict access to your content.

How CloudFront Distribution Works:

1. **User Request:** A user requests content through a URL pointing to your CloudFront distribution.
2. **Edge Location Check:** CloudFront checks if the requested content is already cached at the nearest edge location.
3. **Cache Hit:** If the content is cached (a cache hit), CloudFront serves it directly to the user, reducing latency.
4. **Cache Miss:** If the content is not cached (a cache miss), CloudFront fetches it from the origin, caches it at the edge location, and then serves it to the user.
5. **Content Delivery:** Subsequent requests for the same content will be served from the cache, improving delivery speed.

Benefits of Using CloudFront Distributions:

- **Global Reach:** Deliver content to users from the nearest edge location, reducing latency.
- **Scalability:** Automatically handles varying levels of traffic, from low to very high.
- **Security:** Integrated with AWS Shield, AWS WAF, and SSL/TLS to provide robust security for your content.

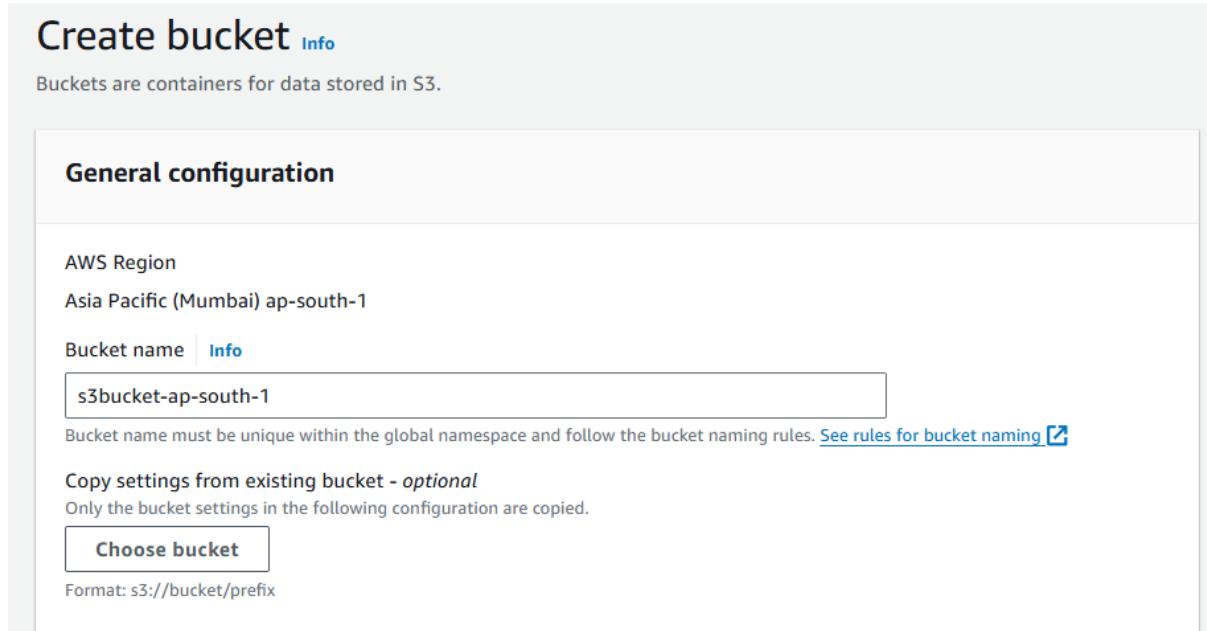
- **Cost Efficiency:** Reduces the load on your origin servers, potentially lowering hosting costs.
- **Customizability:** Fine-tune cache behaviour, security settings, and more to meet specific needs.

CloudFront distributions are essential for delivering content efficiently and securely to a global audience.

In this lab, you'll set up a static website using Amazon S3 and CloudFront. You'll create a public S3 bucket in Mumbai, upload your website, and configure it for hosting. Then, you'll set up CloudFront to distribute your site globally and use Lambda@Edge to direct visitors to the nearest server based on their location. The end goal is to have a globally accessible website that loads efficiently by automatically routing users to the closest server.

To begin with the Lab:

1. In this lab we are going to deploy an Amazon S3 bucket in ap-south-1 which is Mumbai region.
2. Also, we will be hosting a static website on our bucket,
3. Now move to S3 in your console and create your bucket. Give it a name and scroll down. Give your bucket name in this format as shown below.
4. Then you should enable public access to your bucket and create it.



The screenshot shows the 'Create bucket' wizard in the AWS S3 console. The first step, 'General configuration', is selected. It includes fields for 'AWS Region' (set to 'Asia Pacific (Mumbai) ap-south-1') and 'Bucket name' (containing 's3bucket-ap-south-1'). A note below the bucket name field states: 'Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)'.

Below the main form, there's an optional section for 'Copy settings from existing bucket - optional'. It contains a note: 'Only the bucket settings in the following configuration are copied.' and a 'Choose bucket' button. The note also specifies the format: 'Format: s3://bucket/prefix'.

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Block all public access

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

Block public access to buckets and objects granted through new access control lists (ACLs)

S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

Block public access to buckets and objects granted through any access control lists (ACLs)

S3 will ignore all ACLs that grant public access to buckets and objects.

Block public access to buckets and objects granted through new public bucket or access point policies

S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

Block public and cross-account access to buckets and objects through any public bucket or access point policies

S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.



Turning off block all public access might result in this bucket and the objects within becoming public

AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.

I acknowledge that the current settings might result in this bucket and the objects within becoming public.

5. So, you have created your public bucket but we have not updated the policy yet, you are going to update the bucket policy now using the below code.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "PublicReadGetObject",  
      "Effect": "Allow",  
      "Principal": "*",  
      "Action": "s3:GetObject",  
      "Resource": "arn:aws:s3:::s3bucket-ap-south-1/*"  
    }  
  ]  
}
```

6. In our Mumbai bucket we have uploaded our static website and we are going to enable static website hosting.

Amazon S3 > Buckets > s3bucket-ap-south-1

s3bucket-ap-south-1 [Info](#)

Objects | Properties | Permissions | Metrics | Management | Access Points

Objects (1) [Info](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

Name	Type	Last modified	Size	Storage class
index.html	html	September 3, 2024, 10:58:07 (UTC+05:30)	526.0 B	Standard

Edit static website hosting [Info](#)

Static website hosting

Use this bucket to host a website or redirect requests. [Learn more](#)

Static website hosting

- Disable
 Enable

Hosting type

- Host a static website
Use the bucket endpoint as the web address. [Learn more](#)
 Redirect requests for an object
Redirect requests to another bucket or domain. [Learn more](#)

i For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see [Using Amazon S3 Block Public Access](#)

Index document

Specify the home or default page of the website.

index.html

7. Below you can see our website has been hosted successfully. You can get this Index.html file from our GitHub.



Your website is hosted in:

ap-south-1 (DefaultRegionMumbai)

8. We have completed all the work for S3. Now we are going to CloudFront to create our distribution.
9. While creating your distribution you must choose the Mumbai bucket as your default bucket. Then scroll down to the bottom.

Create distribution

Origin

Origin domain

Choose an AWS origin, or enter your origin's domain name.



s3bucket-ap-south-1.s3-website.ap-south-1.amazonaws.com



Protocol Info

- HTTP only
- HTTPS only
- Match viewer

HTTP port

Enter your origin's HTTP port. The default is port 80.

80

10. Here you will find the default root object option, you must write here index.html and create your distribution.

Supported HTTP versions

Add support for additional HTTP versions. HTTP/1.0 and HTTP/1.1 are supported by default.

- HTTP/2
- HTTP/3

Default root object - *optional*

The object (file name) to return when a viewer requests the root URL (/) instead of a specific object.

index.html

Standard logging

Get logs of viewer requests delivered to an Amazon S3 bucket.

- Off
- On

11. The deployment of your distribution might take some time.

CloudFront > Distributions > E2MONHUTSNH778

E2MONHUTSNH778

[View metrics](#)

General Security Origins Behaviors Error pages Invalidations Tags

Details

Distribution domain name d1nj2ggntw7989.cloudfront.net	ARN arn:aws:cloudfront::distribution/E2MONHUTSNH778	Last modified Deploying
---	--	--

- When the status of your CloudFront is enabled, you must copy the domain name and paste it into a new tab. Then you will see your default Mumbai region S3 bucket static hosted website here.

CloudFront > Distributions

Distributions (1) [Info](#)

[Create distribution](#)

Search all distributions

ID	Description	Type	Domain name	Alternate domains	Origins	Status
E2MONHUTSNH778	-	Production	d1nj2ggntw7989.cloudfront.net	-	s3bucket-ap-sou	Enabled

← [C](#) [https://d1nj2ggntw7989.cloudfront.net](#)

Your website is hosted in:

ap-south-1 (DefaultRegionMumbai)

- Now we must create a **lambda@edge function** that will redirect our traffic based on our origin. So, go to lambda and click on the create function. **Just remember to create your lambda function in N. Virginia Region.**
- Here you must choose Use a blueprint and scroll down.

Lambda > Functions > Create function

Create function [Info](#)

Choose one of the following options to create your function.

- Author from scratch
Start with a simple Hello World example.
- Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.
- Container image
Select a container image to deploy for your function.

- Then in the blueprint name search for CloudFront and choose Modify HTTP response header.

cloudfront X

Get started

Modify HTTP response header

Blueprint for modifying **CloudFront** response header implemented in NodeJS.

nodejs18.x

Return HTTP redirect response

Blueprint for returning HTTP redirect implemented in NodeJS.

nodejs18.x

Return HTTP response 200 status code

Blueprint for generating a response from viewer-request trigger implemented in NodeJS.

nodejs18.x

Hello world function

nodejs18.x ▲

A starter AWS Lambda function.

16. Now give your lambda function a name and scroll down.

Basic information Info

Blueprint name

Return HTTP redirect response

nodejs18.x ▼

Blueprint for returning HTTP redirect implemented in NodeJS.

Function name

Enter a name that describes the purpose of your function.

lambdaAWSbucket

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime

nodejs18.x

Architecture

x86_64

17. For the execution role choose to create a new role, then give your role a name and scroll down to the bottom to create your lambda function.

Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

- Create a new role with basic Lambda permissions
- Use an existing role
- Create a new role from AWS policy templates

 Role creation might take a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.

Role name

Enter a name for your new role.

Use only letters, numbers, hyphens, or underscores with no spaces.

Policy templates - *optional* [Info](#)

Choose one or more policy templates.



Basic Lambda@Edge permissions (for CloudFront trigger) 
CloudWatch Logs

18. Then you will see this type of pop-up window, **here you must choose to cancel this window**. We will configure our cloud front and deploy it to **Lambda@edge** later.

Deploy to Lambda@Edge

X



CloudFront
cdn edge

▼

Select an option

- Configure new CloudFront trigger
- Use existing CloudFront trigger on this function

Configure CloudFront trigger

Distribution

The CloudFront distribution that will send events to your Lambda function.

arn:aws:cloudfront::878893308172:distribution/E2M0NHUTSNH778

X

C

Cache behavior

Choose the cache behavior you would like this Lambda function to be associated with.

*

X

C

CloudFront event

Choose one CloudFront event to listen for.

Origin request

▼

Include body

Select "Include body" if you want to read the request body for viewer request or origin request events.

[Learn more](#)

Cancel

Deploy

19. So, follow the below link for AWS documentation and scroll down to this example shown in the snapshot. Copy the code given in this example and paste it into your lambda function.

20. Remember to give your S3 bucket name in the code.

<https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/lambda-examples.html>

Example: Use an origin-request trigger to change the Amazon S3 origin Region

This function demonstrates how an origin-request trigger can be used to change the Amazon S3 origin from which the content is fetched, based on request properties.

In this example, we use the value of the `CloudFront-Viewer-Country` header to update the S3 bucket domain name to a bucket in a Region that is closer to the viewer. This can be useful in several ways:

- It reduces latencies when the Region specified is nearer to the viewer's country.
- It provides data sovereignty by making sure that data is served from an origin that's in the same country that the request came from.

To use this example, you must do the following:

- Configure your distribution to cache based on the `CloudFront-Viewer-Country` header. For more information, see [Cache based on selected request headers](#).
- Create a trigger for this function in the origin request event. CloudFront adds the `CloudFront-Viewer-Country` header after the viewer request event, so to use this example, you must make sure that the function executes for an origin request.

```
'use strict';
```

```
exports.handler = (event, context, callback) => {
  const request = event.Records[0].cf.request;

  const countryToRegion = {
    'SG': 'ap-southeast-1', // Singapore
    'US': 'us-east-1', // North Virginia
    'IN': 'ap-south-1' // India
  };

  if (request.headers['cloudfront-viewer-country']) {
    const countryCode = request.headers['cloudfront-viewer-country'][0].value;
    const region = countryToRegion[countryCode];

    if (region) {
      request.origin.s3.region = region;
      const domainName = `s3bucket-${region}.s3.${region}.amazonaws.com`;
      request.origin.s3.domainName = domainName;
      request.headers['host'] = [{ key: 'host', value: domainName }];
    }
  }
}
```

```
callback(null, request);  
};
```

21. Once you have changed the code then you need to click on deploy the code. After that, the code is deployed and you must click on add trigger in your lambda function.
22. Here you must choose CloudFront. Then click on deploy to Lambda@Edge.

The screenshot shows the 'Add trigger' configuration page for a Lambda function. At the top, there's a search bar with 'Select a source' placeholder and a search input field containing 'cloud'. Below the search results, 'CloudFront' is listed under 'APIs/interactive/web' with icons for 'cdn' and 'edge'. On the right side of the search results, there are 'Cancel' and 'Add' buttons. The main area below the search bar is titled 'Trigger configuration' and contains a section for 'CloudFront' triggers. It includes a description of what a CloudFront trigger does, a 'Deploy a new version of this function to global AWS locations' button, and a note about Lambda@Edge restrictions. At the bottom right are 'Cancel' and 'Add' buttons.

23. Below you can see that it is repopulating our CloudFront behaviour, so choose your distribution and click on deploy.

Deploy to Lambda@Edge

X



CloudFront

cdn edge

Select an option

- Configure new CloudFront trigger
- Use existing CloudFront trigger on this function

Configure CloudFront trigger

Distribution

The CloudFront distribution that will send events to your Lambda function.



arn:aws:cloudfront::878893308172:distribution/E2MONHUTSNH778



Cache behavior

Choose the cache behavior you would like this Lambda function to be associated with.



*



CloudFront event

Choose one CloudFront event to listen for.

Origin request



Include body

Select "Include body" if you want to read the request body for viewer request or origin request events.

[Learn more](#) ↗

Cancel

Deploy

24. Below you can see that this is version 1 of our lambda function.

Lambda > Functions > lambdaedge > Version: 1

Version: 1

[Copy ARN](#) Version: 1 Actions

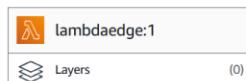
Successfully created version 1 of your function and associated a CloudFront trigger. This function is being replicated across all regions and the associated distribution is being updated as well. This will take a few minutes; go to the [CloudFront console](#) ↗ to monitor distribution status.

▼ Function overview [Info](#)

[Export to Application Composer](#)

[Download](#)

[Diagram](#) [Template](#)



Layers

(0)

Description

Version published for Lambda@Edge

Last modified

33 minutes ago

Function ARN

arn:aws:lambda:us-east-1:878893308172:function:lambdaedge:1



+ Add destination

+ Add trigger

25. Now if you go to CloudFront you can see that it again started to deploy.

[CloudFront](#) > [Distributions](#) > E2MONHUTSNH778

E2MONHUTSNH778

[View metrics](#)

[General](#) | [Security](#) | [Origins](#) | [Behaviors](#) | [Error pages](#) | [Invalidations](#) | [Tags](#)

Details

Distribution domain name d1nj2ggntw7989.cloudfront.net	ARN arn:aws:cloudfront::878893308172:distribution/E2MONHUTSNH778	Last modified Deploying
---	---	----------------------------

26. Now you must wait for some time until your distribution has been deployed. Once it is deployed, then copy the distribution domain name and paste it into a new tab.

[CloudFront](#) > [Distributions](#) > E2MONHUTSNH778

E2MONHUTSNH778

[View metrics](#)

[General](#) | [Security](#) | [Origins](#) | [Behaviors](#) | [Error pages](#) | [Invalidations](#) | [Tags](#)

Details

Distribution domain name d1nj2ggntw7989.cloudfront.net	ARN arn:aws:cloudfront::878893308172:distribution/E2MONHUTSNH778	Last modified September 3, 2024 at 6:36:57 AM UTC
---	---	--

27. Below you can see that using the domain name we can see our website, hosted in our Mumbai S3 bucket.

Your website is hosted in:

ap-south-1 (DefaultRegionMumbai)