

---

# **Amazon CloudFront**

## **Developer Guide**

**API Version 2012-05-05**



## Amazon CloudFront: Developer Guide

Copyright © 2012 Amazon Web Services LLC or its affiliates. All rights reserved.

The following are trademarks or registered trademarks of Amazon: Amazon, Amazon.com, Amazon.com Design, Amazon DevPay, Amazon EC2, Amazon Web Services Design, AWS, CloudFront, EC2, Elastic Compute Cloud, Kindle, and Mechanical Turk. In addition, Amazon.com graphics, logos, page headers, button icons, scripts, and service names are trademarks, or trade dress of Amazon in the U.S. and/or other countries. Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon.

All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Introduction to Amazon CloudFront .....	1
CloudFront Concepts .....	5
How CloudFront Delivers Content .....	7
Locations of CloudFront Edge Servers .....	8
Paying for CloudFront .....	8
CloudFront or Amazon S3? .....	9
Controlling User Access to Your AWS Account .....	10
Getting Started with CloudFront .....	14
Creating Download Distributions .....	21
Creating Streaming Distributions .....	28
Migrating from Amazon S3 to CloudFront .....	32
Working with Distributions .....	34
Changes to the CloudFront API .....	34
Overview of Download and Streaming Distributions .....	35
Actions on Distributions .....	35
Working with Download Distributions .....	36
Working with Streaming Distributions .....	45
Using Alternate Domain Names (CNAMEs) .....	53
Listing, Viewing, and Updating CloudFront Distributions .....	55
Deleting a Distribution .....	59
Working with Objects .....	60
Format of URLs for CloudFront Objects .....	60
Adding, Removing, or Replacing Objects in a Distribution .....	64
Adding Objects to a CloudFront Distribution .....	64
Updating Existing Objects Using Versioned Object Names .....	64
Updating Existing Objects Using the Same Object Names .....	65
Specifying How Long Objects Stay in a CloudFront Edge Cache (Object Expiration) .....	65
Invalidating Objects .....	68
How CloudFront Processes Partial Requests for an Object (Range GETs) .....	71
Specifying a Default Root Object (Download Distributions Only) .....	72
Serving Compressed Files .....	74
Restricting Access to Objects Based on the Geographic Location of End Users (Geoblocking) .....	76
Request and Response Behavior, and Supported HTTP Status Codes .....	77
Request and Response Behavior for Amazon S3 Origins .....	77
Request and Response Behavior, and Supported HTTP Status Codes for Custom Origins .....	79
Using a Signed URL to Serve Private Content .....	84
Overview of Private Content .....	86
How to Serve Private Content Using a Signed URL .....	88
Securing Your Content in Amazon S3 .....	90
Restricting End User Access .....	94
Signature Code, Examples, and Tools .....	110
Create a URL Signature Using Perl .....	110
Create a URL Signature Using PHP .....	111
Create a URL Signature Using C# and the .NET Framework .....	114
Create a URL Signature Using Java .....	122
GUI Tools for Signature Generation .....	125
Using an HTTPS Connection to Access Your Objects .....	126
Access Logs .....	129
General Usage Data .....	138
Troubleshooting .....	139
Making API Requests .....	142
Endpoints .....	142
AWS Support for Programming Languages .....	143
REST Requests .....	143
REST Responses .....	146
Authenticating REST Requests .....	148
CloudFront Tutorials .....	152
Live HTTP Streaming Using CloudFront and Adobe Flash Media Server 4.5 .....	152

Overview of Live HTTP Streaming with Amazon Web Services .....	152
Creating an Amazon Web Services Account .....	153
Creating an Amazon EC2 Key Pair .....	154
Subscribing to Adobe Flash Media Server .....	155
Creating an AWS CloudFormation Stack for Live Streaming .....	155
Verifying that Adobe Flash Media Server Is Running .....	160
Setting Up Adobe Flash Media Live Encoder to Publish a Live Stream .....	161
Embedding Flash Media Playback for an Amazon CloudFront Live HTTP Stream in a Web Application .....	164
Deleting an AWS CloudFormation Stack for Live Streaming .....	166
Frequently Asked Questions .....	167
Additional Documentation .....	172
Live Smooth Streaming Using Amazon CloudFront and IIS Media Services 4.1 .....	174
Overview of Live Smooth Streaming with Amazon Web Services .....	174
Creating an Amazon Web Services Account .....	175
Creating an Amazon EC2 Key Pair .....	175
Creating an AWS CloudFormation Stack for Live Smooth Streaming .....	176
Verifying that Your Amazon EC2 Windows Server Instance Is Running .....	179
Getting Your Windows Password .....	180
Encoding Your Live Stream .....	182
Viewing Your Live Smooth Stream .....	187
Deleting Your AWS CloudFormation Live Smooth Streaming Stack .....	187
Frequently Asked Questions .....	188
Additional Documentation .....	190
Restricting Access to Files in a CloudFront Distribution Based on Geographic Location (Geoblocking) .....	191
Overview of Restricting Access to Files in a CloudFront Distribution Based on Geographic Location .....	191
Creating an Amazon Web Services Account .....	193
Sample Code for Digital Element .....	193
Java Sample Code for Digital Element .....	193
.NET Sample Code for Digital Element .....	197
PHP Sample Code for Digital Element .....	201
Sample Code for MaxMind .....	203
Java Sample Code for MaxMind .....	204
PHP Sample Code for MaxMind .....	208
.NET Sample Code for MaxMind .....	210
Frequently Asked Questions .....	213
Additional Services and Documentation .....	214
Amazon CloudFront Resources .....	216
Where Do I Go from Here? .....	218
Document History .....	222

# Introduction to Amazon CloudFront

---

## Topics

- [What Is Amazon CloudFront and Why Do I Need It? \(p. 1\)](#)
- [How Do I...? \(p. 3\)](#)
- [CloudFront Concepts \(p. 5\)](#)
- [How CloudFront Delivers Content \(p. 7\)](#)
- [Locations of CloudFront Edge Servers \(p. 8\)](#)
- [Paying for CloudFront \(p. 8\)](#)
- [CloudFront or Amazon S3? \(p. 9\)](#)
- [Controlling User Access to Your AWS Account \(p. 10\)](#)

## What Is Amazon CloudFront and Why Do I Need It?

CloudFront is a web service that speeds up distribution of your static and dynamic web content, for example, .html, .css, .php, and image files, to end users. CloudFront delivers your content through a worldwide network of edge locations. When an end user requests content that you're serving with CloudFront, the user is routed to the edge location that provides the lowest latency, so content is delivered with the best possible performance. If the content is already in that edge location, CloudFront delivers it immediately. If the content is not currently in that edge location, CloudFront retrieves it from an Amazon S3 bucket or an HTTP server (for example, a web server) that you have identified as the source for the definitive version of your content.

This concept is best illustrated by an example. Suppose you're serving the following image from a traditional web server, not from CloudFront:



(The image is owned by NASA and comes from the Visible Earth website, <http://visibleearth.nasa.gov/>.)

You're serving the image using the URL `http://example.com/globe_west_540.jpg`. Your end users can easily navigate to this URL and see the image, but they probably don't know that their request was routed from one network to another—through the complex collection of interconnected networks that comprise the Internet—until the image was found.

Further suppose that the web server from which you're serving the image is in Seattle, Washington, USA, and that an end user in Austin, Texas, USA requests the image. The traceroute list below (courtesy of [www.WatchMouse.com](http://www.WatchMouse.com)) shows one way that this request could be routed.

1	vrid-225.core-sw.aus.us.siteprotect.com (216.139.225.1) 0.627 ms
2	xe-3-4.brdr-rtr-02.aus.us.siteprotect.com (216.139.253.53) 0.219 ms
3	66.113.197.121 0.452 ms
4	xe-5-2-0.edge3.Dallas1.Level3.net (4.59.112.37) 4.978 ms
5	ae-73-70.ebr3.Dallas1.Level3.net (4.69.145.116) 9.817 ms
6	ae-7-7.ebr3.Atlanta2.Level3.net (4.69.134.22) 30.570 ms
7	ae-2-2.ebr1.Washington1.Level3.net (4.69.132.86) 38.801 ms
8	ae-81-81.csw3.Washington1.Level3.net (4.69.134.138) 41.795 ms
9	ae-3-89.edge2.Washington1.Level3.net (4.68.17.145) 39.193 ms
10	72.21.222.139 35.767 ms



In this example, the request was routed 10 times within the United States before the image was retrieved, which is not an unusually large number of hops. If your end user were in Europe, the request would be routed through even more networks to reach your server in Seattle. The number of networks and the distance that the request and the image must travel have a significant impact on the performance, reliability, and availability of the image.

CloudFront speeds up the distribution of your content by routing end users to the nearest CloudFront edge location (in terms of latency) in a worldwide network of edge locations, which dramatically reduces the number of networks that your users' requests must pass through. This improves performance: end users get lower latency (the time it takes to load the first byte of the object) and higher data transfer rates. You also get increased reliability and availability because there is no longer a central point of failure—copies of your object are now held in edge locations around the world.

For a list of the locations of CloudFront edge servers, see [The Amazon CloudFront Edge Network](#) on the CloudFront detail page.

## How Do I...?

How Do I?	Relevant Topics
Get Started	<a href="#">Getting Started with CloudFront (p. 14)</a>
Learn if CloudFront is right for my use case	<a href="#">Introduction to Amazon CloudFront (p. 1)</a>
Use CloudFront distributions	<a href="#">Working with Distributions (p. 34)</a>

How Do I?	Relevant Topics
Create a download distribution	<a href="#">Creating Download Distributions (p. 21)</a>
Create a streaming distribution	<a href="#">Creating Streaming Distributions (p. 28)</a>
Migrate from Amazon S3 to CloudFront	<a href="#">Migrating from Amazon S3 to CloudFront (p. 32)</a>
Understand the security protocols used with CloudFront	<a href="#">Using an HTTPS Connection to Access Your Objects (p. 126)</a>
Use the CloudFront API	<a href="#">Making API Requests (p. 142)</a>



# CloudFront Concepts

This section explains the basic concepts behind CloudFront.

## Objects

Objects are the files you want CloudFront to deliver. Objects typically include web pages, images, and media files, but can be anything that can be served over HTTP or one of the versions of Adobe RTMP, the protocol used by Adobe Flash Media Server.

## Origin Server

An origin server is the location where you store the original, definitive version of your objects. If you're serving content over HTTP, your origin server is either an Amazon S3 bucket or an HTTP server, for example, a web server. Your HTTP server can be running on an Amazon Elastic Compute Cloud instance or on a server that you manage; these servers are also known as *custom origins*.

If you're serving media files using RTMP, your origin server is always an Amazon S3 bucket.

## Distributions

After you store your objects in your origin server, you create a distribution, which tells CloudFront where your objects are. When you create a distribution, CloudFront gives you a unique domain name that you use to reference your objects, for example, d111111abcdef8.cloudfront.net.

Suppose your origin is an Amazon S3 bucket called myawsbucket that contains an image called `images/image.jpg`. You want to display the image on your web page. If you display the image directly from your Amazon S3 bucket, the link is:

```
http://myawsbucket.s3.amazonaws.com/images/image.jpg
```

If you instead display the image using CloudFront the link is:

```
http://myawsbucket.cloudfront.net/images/image.jpg
```

If you already have a domain name (for example, `example.com`) that you want to use for links to your objects, you can specify up to 10 alternate domain names, also known as CNAMEs. If you add a alternate domain name for `example.com`, the link is:

```
http://example.com/images/image.jpg
```

For more information about alternate domain names, see [Using Alternate Domain Names \(CNAMEs\)](#) (p. 53).

You can create two types of distributions:

- A **download distribution** delivers content using HTTP or HTTPS. Using a download distribution, you can configure CloudFront to access your web content in any combination of up to 10 Amazon S3 buckets and custom origins.
- A **streaming distribution** delivers digital media using Adobe Flash Media Server and the Real-Time Messaging Protocol. The origin for a streaming distribution is always one Amazon S3 bucket.

## Edge Locations

An edge location is a geographical site where CloudFront caches copies of your objects. When an end user requests one of your objects, CloudFront decides which edge location is best able to serve the request. If that edge location doesn't have the object, CloudFront gets a copy from the origin server, serves it to the end user, and stores it in the cache at that edge location.

For the locations of CloudFront edge locations, go to the [CloudFront Details](#) page.

## Expiration

By default, an object expires after being in an edge location for 24 hours. After the object expires, the next time CloudFront gets a request for the object, the request is forwarded to your origin to determine whether an updated version is available. If you have updated the object on your origin server, the origin server returns the updated version. CloudFront serves it to the end user and stores the updated version in the cache at that edge location. The minimum expiration time is 0 seconds; there isn't a maximum expiration time limit. For more information about expiration, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Object Expiration\)](#) (p. 65).

## Eventual Consistency

When you create, modify, or delete a CloudFront distribution, it takes time for your changes to propagate to the CloudFront database. Information about the distribution is eventually consistent, but an immediate request for information about a distribution might not show the change. Consistency is usually reached within minutes, but a high system load or network partition might increase this time.

# How CloudFront Delivers Content

Here is an overview of how CloudFront delivers your content, including the steps you perform before your first end user accesses your application or website. Let's assume that you're using a download distribution to deliver image files from an Amazon S3 bucket and to server HTML content from an HTTP server. The process for serving media files using a streaming distribution is very similar.

Let's also assume that you make the objects in your bucket publicly readable, so anyone who knows the CloudFront URLs for your objects can access them. If you want to keep the objects private and control who accesses them, see [Using a Signed URL to Serve Private Content \(p. 84\)](#).

## Process for Delivering Content

1	<p>You configure your origin servers, which are the servers from which CloudFront gets your files for distribution at edge locations all over the world. For this example:</p> <ul style="list-style-type: none"><li>• You create an Amazon S3 bucket for your image files, and you make the files publicly available.</li><li>• You configure an HTTP server for your HTML content.</li></ul>
2	<p>You upload your files to your origin servers.</p>
3	<p>You create a CloudFront distribution, which tells CloudFront which origin servers to get your files from when end users request the files through your web site or application. For this example, you configure your distribution so CloudFront gets your image files from one server and PHP files from another.</p> <p>As you create your distribution, you also specify details such as whether you want CloudFront to log all requests and whether you want the distribution to be enabled as soon as it's created.</p>
4	<p>CloudFront returns the domain name that you use for your objects as you develop your web site or application. For example, if CloudFront returns <code>d1234.cloudfront.net</code> as the domain name for your distribution, the URL for <code>image.jpg</code> in your Amazon S3 bucket (or in the root directory on an HTTP server) will be <code>http://d1234.cloudfront.net/image.jpg</code>.</p> <p>You can also configure your CloudFront distribution so you can use your own domain name. In that case, the URL might be <code>http://www.example.com/image.jpg</code>.</p>
5	<p>CloudFront distributes information about your distribution to all of its edge locations.</p>
6	<p>You develop your website or application to use the domain name that CloudFront returned in Step 4 or to use your own domain name, depending on how you configured your distribution.</p> <p>You can optionally configure your origin server to add headers to the files; the headers indicate how long you want the files to stay in the cache in CloudFront edge locations. By default, each object stays in an edge location for 24 hours.</p>
7	<p>An end user accesses your website or application and requests an image file and an HTML file.</p>
8	<p>CloudFront determines which edge location can best serve the end user's request. In this example, let's say it's Tokyo.</p>

9	<p>In the Tokyo edge location, CloudFront checks its cache for the requested files. If the files are in the cache, CloudFront returns them to the end user. If the files are <i>not</i> in the cache:</p> <ul style="list-style-type: none"><li>• CloudFront compares the request with the specifications in your distribution and forwards the request for the files to the applicable origin server for the corresponding file type: to your Amazon S3 bucket for image files and to your HTTP server for the HTML files.</li><li>• The origin servers send the files back to the CloudFront edge location in Tokyo.</li><li>• CloudFront forwards the files to the end user and adds the files to the cache in Tokyo for the next time someone requests those files.</li></ul>
10	<p>After an object has been in an edge cache for 24 hours or for the duration specified in your file headers (see Step 6), CloudFront forwards the next request for the object to your origin to determine whether the edge location has the latest version. If the version in the edge location is the latest, CloudFront delivers it to your end user. If not, your origin sends the latest version to CloudFront, and CloudFront delivers the object to your end user and stores the latest version in the cache at that edge location. For more information, see <a href="#">Specifying How Long Objects Stay in a CloudFront Edge Cache (Object Expiration)</a> (p. 65).</p>

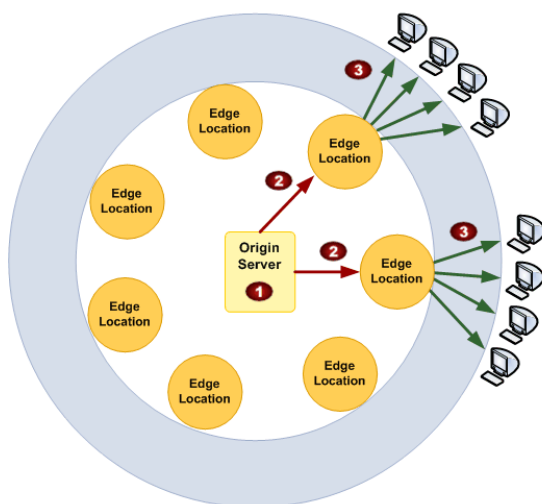
## Locations of CloudFront Edge Servers

For a list of the locations of CloudFront edge servers, see [The Amazon CloudFront Edge Network](#) on the Amazon CloudFront detail page.

## Paying for CloudFront

CloudFront is designed so you don't have to pay any upfront fees or commit to how much content you'll have. As with the other AWS services, you pay as you go and pay only for what you use.

The following diagram and table summarize the charges to use CloudFront.



Your monthly bill from AWS separates your usage and dollar amounts by AWS service and function. As a result, you see some charges for storing objects with Amazon S3 (1) (if you are using Amazon S3 as your origin server), some charges for data transfer between your bucket and your edge location (2), and some charges for serving data from CloudFront (3).

	Charge	Comments
1	Storage in an Amazon S3 origin server	You pay normal Amazon S3 storage charges to store objects in your bucket; the charges appear in the Amazon S3 portion of your AWS statement.
2	Copying objects to edge locations	<p>If using an Amazon S3 origin server, you incur the normal Amazon S3 charges for GET requests and for data transfer out. CloudFront copies an object to an edge location only if there is demand for that object at that edge location.</p> <p>The data transfer charges appear in the AWS Data Transfer portion of your AWS statement.</p>
3	Serving objects from edge locations	You incur CloudFront charges for requests and data transfer out, which are lower than the corresponding Amazon S3 charges. The CloudFront charges appear in the CloudFront portion of your AWS statement. For more information, see <a href="#">Amazon CloudFront Pricing</a> .

## CloudFront or Amazon S3?

Both CloudFront and Amazon S3 serve content. Should you use CloudFront to serve all your content? Not necessarily. It depends on your particular needs.

Amazon S3 is designed to store the original, definitive version of your files. It is optimized for high durability and cost-effective application storage and data transfer.

CloudFront is designed to distribute your most popular content with low latency. It is not designed for durable storage. Copies of your popular objects are stored in edge locations close to end users on the Internet; if an object isn't accessed frequently it might be removed from an edge location. For objects that are served many times, CloudFront can lower the cost of delivery while providing a faster download experience.

If you expect a large number of requests for each of your files, CloudFront provides higher performance than Amazon S3 alone, because objects are stored closer to end users' locations. You might also find CloudFront a more cost-effective choice than Amazon S3 for delivery of popular objects due to its lower charges for data transfer at higher usage tiers.

# Controlling User Access to Your AWS Account

## Topics

- [CloudFront Resources](#) (p. 10)
- [CloudFront Actions](#) (p. 10)
- [Policy Keys](#) (p. 11)
- [Example Policies for CloudFront](#) (p. 11)

Amazon CloudFront integrates with AWS Identity and Access Management (IAM) so that you can create Users for your AWS Account and you can specify which CloudFront actions a User (or a group of Users) can perform in your AWS Account. You control User access to CloudFront by creating policies that describe User or group permissions. For example, you might create a policy that gives only certain Users in your organization permission to use `GetDistributionConfig`. They could then use the action to retrieve data about your CloudFront distributions.

For more information on using policies to set AWS Account User permissions, go to [Permissions and Policies](#) in *Using AWS Identity and Access Management*. For general information about IAM, go to [AWS Identity and Access Management](#) on the AWS website.

## Important

Using Amazon CloudFront with IAM doesn't change how you use CloudFront. There are no changes to CloudFront actions, and no new CloudFront actions related to Users and access control.

## CloudFront Resources

You use an asterisk (\*) as the resource when writing a policy to control access to CloudFront actions. This is because you can't use IAM to control access to specific CloudFront resources. For example, you can't give Users access to a specific distribution. Permissions granted using IAM include all the resources you use with CloudFront. Because you cannot specify the resources to control access to, there are no CloudFront resource ARNs (Amazon Resource Names) for you to use in an IAM policy. (For detailed information about using ARNs with IAM, go to "ARNs" in the [Identifiers for IAM Entities](#) section of *Using AWS Identity and Access Management*.)

## CloudFront Actions

In an IAM policy, you can specify any and all API actions that CloudFront offers. The action name must be prefixed with the lowercase string `cloudfront:`. For example:  
`cloudfront:GetDistributionConfig`, `cloudfront:ListInvalidations`, or `cloudfront:*` (for all CloudFront actions).

The following tables list the canonical names for all CloudFront actions. Use these canonical names when specifying APIs in IAM policies.

API Actions for Download Distributions	Canonical Name
<a href="#">POST Distribution</a>	CreateDistribution
<a href="#">GET Distribution</a>	GetDistribution
<a href="#">GET Distribution Config</a>	GetDistributionConfig
<a href="#">PUT Distribution Config</a>	UpdateDistribution

API Actions for Download Distributions	Canonical Name
<code>GET Distribution List</code>	ListDistributions
<code>DELETE Distribution</code>	DeleteDistribution

API Actions for Streaming Distributions	Canonical Name
<code>POST Streaming Distribution</code>	CreateStreamingDistribution
<code>GET Streaming Distribution</code>	GetStreamingDistribution
<code>GET Streaming Distribution Config</code>	GetStreamingDistributionConfig
<code>PUT Streaming Distribution Config</code>	UpdateStreamingDistribution
<code>GET Streaming Distribution List</code>	ListStreamingDistributions
<code>DELETE Streaming Distribution</code>	DeleteStreamingDistribution

API Actions for Invalidations	Canonical Name
<code>POST Invalidation</code>	CreateInvalidation
<code>GET Invalidation</code>	GetInvalidation
<code>GET Invalidation List</code>	ListInvalidations

API Action for Origin Access Identities	Canonical Name
<code>POST Origin Access Identity</code>	CreateCloudFrontOriginAccessIdentity
<code>GET Origin Access Identity</code>	GetCloudFrontOriginAccessIdentity
<code>GET Origin Access Identity Config</code>	GetCloudFrontOriginAccessIdentityConfig
<code>PUT Origin Access Identity Config</code>	UpdateCloudFrontOriginAccessIdentity
<code>GET Origin Access Identity List</code>	ListCloudFrontOriginAccessIdentities
<code>DELETE Origin Access Identity</code>	DeleteCloudFrontOriginAccessIdentity

## Policy Keys

Policy keys enable you to add conditions to your policies, such as request date or IP range. CloudFront implements the AWS-wide policy keys, but no others. For more information about policy keys, see "Condition" in the [Element Descriptions](#) section of *Using AWS Identity and Access Management*.

## Example Policies for CloudFront

This section shows a few simple policies for controlling user access to CloudFront.

## Note

In the future, CloudFront might add new actions that should logically be included in one of the following policies, based on the policy's stated goals.

### Example 1: Allow a group read and write access to all of resources owned by the account

This example creates a policy that is attached to a group (for example, the Developers group) to give the group read and write access to all CloudFront resources.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": ["cloudfront:*"],
    "Resource": "*"
  }]
}
```

### Example 2: Allow a group read access to all of resources owned by the account

This example creates a policy that is attached to a group (for example, the Finance group) to give the group read access to all CloudFront resources.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": ["cloudfront:Get*", "cloudfront:List*"],
    "Resource": "*"
  }]
}
```

### Example 3: Allow a group read and write access to all distributions owned by the account

This example creates a policy that is attached to a group (for example, the Ops group) to give the group read and write access to all distributions, but not access to invalidations or origin access identities.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": ["cloudfront:*Distribution*"],
    "Resource": "*"
  }]
}
```



**Example 4: Allow a group to retrieve CloudFront distribution data, but only if they're using SSL with the request**

This example creates a policy that is attached to a group to give the group access to all CloudFront actions, with a condition that requires use of SSL.

```
{
  "Statement": [{
    "Effect": "Allow",
    "Action": ["cloudfront:*"],
    "Resource": "*",
    "Condition": {
      "Bool": {
        "aws:SecureTransport": "true"
      }
    }
  }]
}
```

# Getting Started with CloudFront

---

The example in this topic gives you a quick overview of how to use CloudFront to:

- Store the original versions of your objects in one Amazon Simple Storage Service (Amazon S3) bucket.
- Distribute download content such as text or graphics.
- Make your objects accessible to everyone.
- Use the CloudFront domain name in URLs for your objects (for example, `http://d1111111abcdef8.cloudfront.net.cloudfront.net/image.jpg`) instead of your own domain name (for example, `http://www.example.com/image.jpg`).
- Keep your objects in CloudFront edge locations for the default duration of 24 hours. (The minimum duration is 0 seconds.)

For information about how to use CloudFront when you want to use other options, see [Creating Download Distributions \(p. 21\)](#) or [Creating Streaming Distributions \(p. 28\)](#).

You only need to perform a few basic steps to start delivering your content using CloudFront. The first step is signing up. After that, you create a CloudFront distribution, and then use the CloudFront domain name to reference content in your web pages or applications.

## Step 1: Sign up for Amazon Web Services

If you haven't already done so, sign up for Amazon Web Services at <http://aws.amazon.com>. Just click **Sign Up Now** and enter any required information.

## Step 2: Upload your content to Amazon S3 and grant object permissions

An Amazon S3 bucket is a container that can contain objects or folders. CloudFront can distribute almost any type of object for you using an Amazon S3 bucket as the source, for example, text, images, and videos. You can create multiple buckets, and there is no limit to the amount of data that you can store on Amazon S3.

## Amazon CloudFront Developer Guide

### Step 2: Upload your content to Amazon S3 and grant object permissions

---

By default, your Amazon S3 bucket and all of the objects in it are private—only the AWS account that created the bucket has permission to read or write the objects in it. If you want to allow anyone to access the objects in your Amazon S3 bucket using CloudFront URLs, you must grant public read permissions to the objects. (This is one of the most common mistakes when working with CloudFront and Amazon S3. You must explicitly grant privileges to each object in an Amazon S3 bucket.)

#### Note

If you want to restrict who can download your content, you can use the CloudFront private content feature. For more information about distributing private content, see [Using a Signed URL to Serve Private Content \(p. 84\)](#).

#### To upload your content to Amazon S3 and grant read permission to everyone

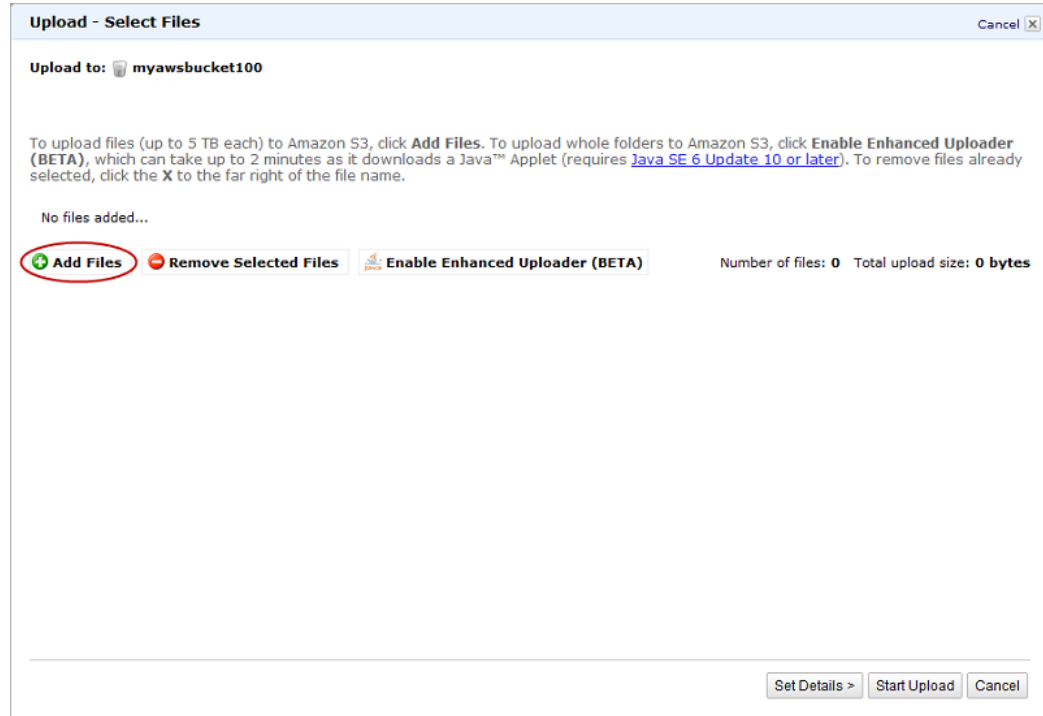
1. Sign in to the AWS Management Console and open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. If you don't have an Amazon S3 bucket yet, go to Step 3 to create one.

If you already have an Amazon S3 bucket but the objects that you want to distribute using CloudFront aren't in it yet, go to Step 6.

If your objects are already in an Amazon S3 bucket, perform the following steps to make them publicly available.

- a. In the **Buckets** pane, select your bucket.
  - b. In the **Objects and Folders** pane, select all of the objects that you want to distribute using CloudFront.
  - c. Click **Actions**, and click **Make Public**.
  - d. Click **OK** to confirm.
  - e. Skip the rest of this procedure, and go to [Step 3: Create a CloudFront Download Distribution \(p. 16\)](#).
3. In the Amazon S3 console, click **Create Bucket**.
  4. In the **Create Bucket** dialog:
    - a. Enter a bucket name. Your bucket name must be all lowercase, and cannot contain any spaces.

Amazon S3 bucket names must be globally unique, just like domain names: no two buckets can have the same name, even if they're owned by different customers. For example, you could create a bucket name by combining the name of your company (the Example Company) with the name of your company project (the myawsbucket project): example-myawsbucket. You can choose any name as long as it's not already in use by another bucket.
    - b. Select a region for your bucket. By default, Amazon S3 creates buckets in the US-Standard region. We recommend that you choose a region close to you to optimize latency, minimize costs, or to address regulatory requirements.
  5. Click **Create**.
  6. Select your bucket in the **Buckets** pane, and click **Upload**.
  7. On the **Upload - Select Files** page, click **Add Files**, and choose the files that you want to upload.



8. Enable public read privileges for each object that you upload to your Amazon S3 bucket.
  - a. Click **Set Details**.
  - b. On the **Set Details** page, click **Set Permissions**.
  - c. On the **Set Permissions** page, click **Make everything public**.
9. Click **Start Upload**.

After the upload completes, you can navigate to this item by its URL. In the case of the previous example, the URL would be:

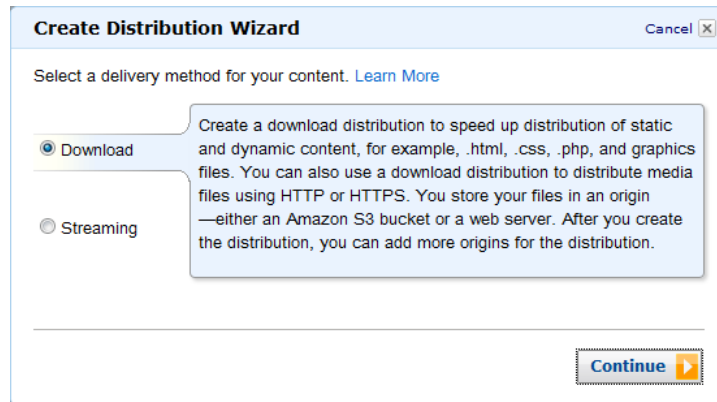
```
http://s3.amazonaws.com/example-myawsbucket/[file name]
```

Use your Amazon S3 URL to verify that your content is publicly accessible, but remember that this is not the URL you will use when you are ready to distribute your content.

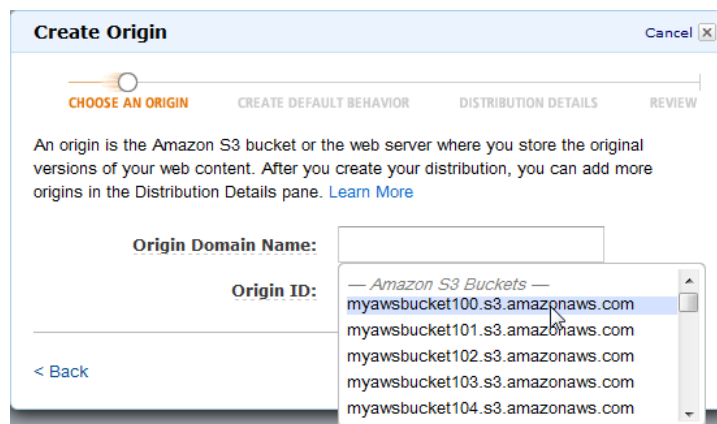
## Step 3: Create a CloudFront Download Distribution

To create a CloudFront download distribution

1. Open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Click **Create Distribution**.
3. On the first page of the **Create Distribution Wizard**, accept the default selection, **Download**, and click **Continue**.



- On the **Create Origin** page, select the Amazon S3 bucket that you created earlier.



- For the value of **Origin ID**, accept the default value and click **Continue**.
- On the **Create Default Cache Behavior** page, accept the default values, and CloudFront will:
  - Forward all requests that use the CloudFront URL for your distribution (for example, `http://d1l1ll1llabcdef8.cloudfront.net.cloudfront.net/image.jpg`) to the Amazon S3 bucket that you specified in Step 4.
  - Allow end users to use either HTTP or HTTPS to access your objects.
  - Cache your objects at CloudFront edge locations for 24 hours.
  - Exclude query string parameters, if any, when forwarding requests for objects to your origin.

For more information about cache behavior options, see [Cache Behaviors \(p. 41\)](#).

**Create Cache Behavior** Cancel

CHOOSE AN ORIGIN   CREATE DEFAULT BEHAVIOR   DISTRIBUTION DETAILS   REVIEW

A cache behavior determines how CloudFront communicates with your origin. After you create your distribution, you can add more cache behaviors in the Distribution Details pane. [Learn More](#)

**Path Pattern:** Default (\*)

**Origin:** S3-myawsbucket100

**Viewer Protocol Policy:** ☒ HTTP and HTTPS ☐ HTTPS Only

**Object Caching:** ☒ Use Origin Cache Headers ☐ Customize

**Minimum TTL:** 0

**Forward Query Strings:** ☐ Yes ☒ No (Improves Caching)

[< Back](#) Continue >

7. On the **Distribution Details** page, enter the applicable values:

- **Alternate Domain Names (CNAMEs):** Optional. Specify one or more domain names that you want to use for URLs for your objects instead of the domain name that CloudFront assigns when you create your distribution. For example, if you want the URL for the object:

`images/image.jpg`

to look like this:

`http://www.example.com/images/image.jpg`

instead of like this:

`http://d111111abcdef8.cloudfront.net/images/image.jpg`

you would create a CNAME for `www.example.com`. You can create up to 10 CNAMEs per distribution.

### Note

If you add a CNAME for `www.example.com` to your distribution, you also need to create (or update) a CNAME record with your DNS service to route queries for `www.example.com` to `d111111abcdef8.cloudfront.net`. For more information, see [Using Alternate Domain Names \(CNAMEs\)](#) (p. 53).

### Important

You must own any domain name that you specify as an alternate domain name.

- **Default Root Object:** Optional. The object that you want CloudFront to request from your origin (for example, `index.html`) when a viewer requests the root URL of your distribution (`http://www.example.com/`) instead of an object in your distribution (`http://www.example.com/product-description.html`). Specifying a default root object avoids exposing the contents of your distribution.
- **Logging:** Optional. If you want CloudFront to log information about each request for an object and store the log files in an Amazon S3 bucket, select **On**, and specify the bucket and an optional prefix for the names of the log files. There is no extra charge to enable logging, but you accrue the usual Amazon S3 charges for storing and accessing the files. CloudFront doesn't delete the logs automatically, but you can delete them at any time.

- **Comments:** Optional. Enter any comments that you want to save with the distribution.
- **Distribution State:** Select **Enabled** if you want CloudFront to begin processing requests as soon as the distribution is created, or select **Disabled** if you do not want CloudFront to begin processing requests after the distribution is created.

The screenshot shows the 'Create Distribution' wizard in the AWS Management Console, specifically the 'Distribution Details' step. The wizard has four steps: 'CHOOSE AN ORIGIN', 'CREATE DEFAULT BEHAVIOR', 'DISTRIBUTION DETAILS', and 'REVIEW'. The 'DISTRIBUTION DETAILS' step is currently active. It contains the following fields and options:

- Alternate Domain Names (CNAMEs):** A text input field with a placeholder and a help icon. Below it, a message says 'Additional configuration is required.' with a question mark icon.
- Default Root Object:** A text input field.
- Logging:** Radio buttons for 'On' and 'Off', with 'Off' selected.
- Bucket for Logs:** A text input field.
- Log Prefix:** A text input field.
- Comments:** A text input field.
- Distribution State:** Radio buttons for 'Enabled' and 'Disabled', with 'Enabled' selected.

At the bottom of the form, there is a '< Back' button and a 'Continue >' button with a right arrow icon.

8. Click **Continue**.
9. On the **Review** page, review your settings, and click **Create Distribution**.
10. After CloudFront has created your distribution, the value of the **Status** column for your distribution will change from **InProgress** to **Deployed**. If you chose to enable the distribution in Step 7, it will then be ready to process requests. This should take less than 15 minutes.

The domain name that CloudFront assigns to your distribution appears in the list of distributions. (It also appears on the **General** tab for a selected distribution.)

## Step 4: Test your links

After you've created your distribution, CloudFront knows where your Amazon S3 origin server is, and you know the domain name associated with the distribution. You can create a link to your Amazon S3 bucket content with that domain name, and have CloudFront serve it.

### Note

You must wait until the status of your distribution changes to **Deployed** before testing your links.

### To link to your objects

1. Copy the following HTML into a new file:
  - Replace <domain name> with the domain name that CloudFront assigned to your distribution.
  - Replace <object name> with the name of a file in your Amazon S3 bucket.

```
<html>
<head>My CloudFront Test</head>
<body>
<p>My text content goes here.</p>
<p> 
</body>
</html>
```

For example, if your domain name was `d1111111abcdef8.cloudfront.net` and your object was `image.jpg`, the URL for the link would be:

`http://d1111111abcdef8.cloudfront.net/image.jpg`.

If your object is in a folder within your bucket, include the folder in the URL. For example, if `image.jpg` is located in an `images` folder, then the URL would be:

`http://d1111111abcdef8.cloudfront.net/images/image.jpg`

2. Save the text in a file that has a `.html` filename extension.
3. Open your web page in a browser to ensure that you can see your content. If you cannot see the content, confirm that you have performed all of the steps correctly. You can also see the tips in [Troubleshooting \(p. 139\)](#).

The browser returns your page with the embedded image file, served from the edge location that CloudFront determined was appropriate to serve the object.

For more information on using CloudFront, go to [Where Do I Go from Here? \(p. 218\)](#).



# Creating Download Distributions

---

This section describes how to create a CloudFront download distribution. For more information about download distributions, see [Working with Download Distributions \(p. 36\)](#).

## Process for Creating a Download Distribution

The following table summarizes the process for creating a download distribution.

### To Create a Download Distribution

1	<p>Create one or more Amazon S3 buckets or configure HTTP servers as your origin servers. An origin is the location where you store the original version of your web content. When CloudFront gets a request for your files, it goes to the origin to get the files that it distributes at edge locations. You can use any combination of up to 10 Amazon S3 buckets and HTTP servers as your origin servers.</p> <p>If you're using an Amazon EC2 server or another custom origin, review <a href="#">Requirements and Recommendations for Using Amazon EC2 and Other Custom Origins (p. 45)</a>.</p>
2	<p>Upload your content to your origin servers. If you don't want to restrict access to your content using CloudFront signed URLs, make the objects publicly readable.</p> <p><b>Caution</b></p> <p>You are responsible for ensuring the security of your origin server. You must ensure that CloudFront has permission to access the server and that the security settings are appropriate to safeguard your content.</p>
3	<p>Create your CloudFront download distribution:</p> <ul style="list-style-type: none"><li>• For more information about creating a download distribution using the CloudFront console, see <a href="#">Creating a Download Distribution Using the CloudFront Console (p. 22)</a>.</li><li>• For information about creating a download distribution using the CloudFront API, go to <a href="#">POST Distribution</a> in the <i>Amazon CloudFront API Reference</i>.</li></ul>

4	Optional: If you created your distribution using the CloudFront console, create more cache behaviors or origins for your distribution. For more information, see <a href="#">To List, View, and Update CloudFront Distributions Using the CloudFront Console</a> (p. 56).
5	Test your download distribution. For more information, see <a href="#">Testing Your Download Distribution</a> (p. 26).
6	<p>Develop your website or application to access your content using the domain name that CloudFront returned after you created your distribution in Step 3. For example, if CloudFront returns <code>d11111abcdef8.cloudfront.net</code> as the domain name for your distribution, the URL for the file <code>image.jpg</code> in an Amazon S3 bucket or in the root directory on an HTTP server will be <code>http://d11111abcdef8.cloudfront.net/image.jpg</code>.</p> <p>If you specified one or more alternate domain names (CNAMEs) when you created your distribution, you can use your own domain name. In that case, the URL for <code>image.jpg</code> might be <code>http://www.example.com/image.jpg</code>.</p> <p>Note the following:</p> <ul style="list-style-type: none"><li>• If you want to use signed URLs to restrict access to your content, see <a href="#">Using a Signed URL to Serve Private Content</a> (p. 84).</li><li>• If you want to serve compressed content, see <a href="#">Serving Compressed Files</a> (p. 74).</li><li>• For information about CloudFront request and response behavior for Amazon S3 and custom origins, see <a href="#">Request and Response Behavior, and Supported HTTP Status Codes</a> (p. 77).</li></ul>

## Creating a Download Distribution Using the CloudFront Console

The following procedure walks you through creating a download distribution using the CloudFront console.

### Note

A few CloudFront settings are currently not available in the CloudFront console:

- Specifying the AWS accounts that are allowed to sign URLs for private content. For more information, see [Requiring Signed URLs](#) (p. 95).
- Creating an origin access identity, which prevents users from accessing files in an Amazon S3 bucket by using the Amazon S3 URL. For more information, see [Securing Your Content in Amazon S3](#) (p. 90).
- Associating an origin access identity with your CloudFront distribution. For more information, see [Creating a Private Content Distribution](#) (p. 92).

If you want to use these settings, you must create the download distribution using the CloudFront API. For information about creating a download distribution using the CloudFront API, go to [POST Distribution](#) in the *Amazon CloudFront API Reference*.

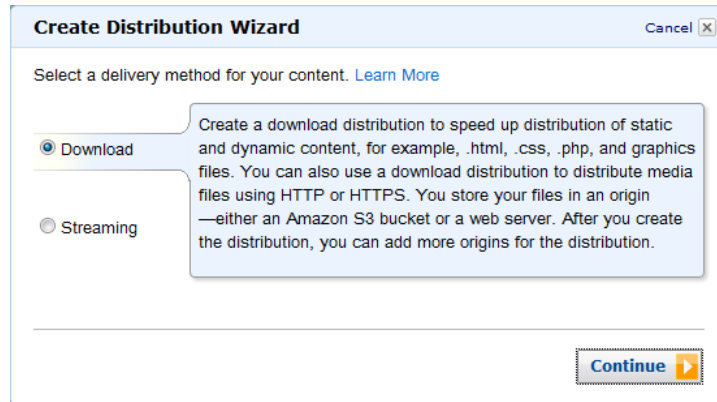
### To create a CloudFront download distribution using the CloudFront console

1. Sign in to the AWS Management Console and open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Click **Create Distribution**.

## Amazon CloudFront Developer Guide

### Creating a Download Distribution Using the CloudFront Console

- On the first page of the **Create Distribution Wizard**, accept the default selection, **Download**, and click **Continue**.

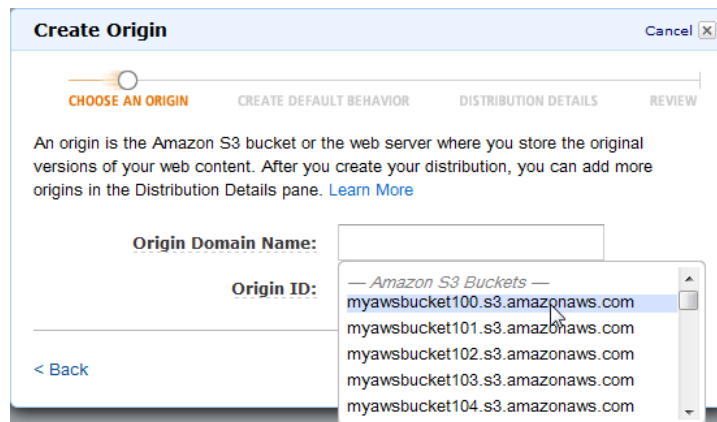


- If you're using an HTTP server as the origin for your distribution, skip to Step 6.

If you're using an Amazon S3 bucket as the origin and if the bucket is associated with the AWS account that you're logged on with, on the **Choose an Origin** page, click in the **Origin Domain Name** field, and select the bucket from the list.

If you're using a bucket that is associated with another AWS account, enter the bucket name in the following format:

*bucket-name*.s3.amazonaws.com



- In the **Origin ID** field, enter a string that will help you distinguish this origin from other origins in this distribution. If you add more origins and cache behaviors to the distribution later in the process, you will link origins with cache behaviors using the origin ID.

Then click **Continue** and skip to Step 7.

- If you're using an HTTP server as the origin for your distribution, specify the following information:

- Origin Domain Name:** Enter the domain name of your HTTP server, for example, `www.example.com`.
- Origin ID:** A string that will help you distinguish this origin from other origins in this distribution. If you add more origins and cache behaviors to the distribution later in the process, you will link origins with cache behaviors using the origin ID.

The following fields appear after you enter a domain name. Enter the applicable values:

- **Origin Protocol Policy:** The protocol policy that you want CloudFront to use when fetching objects from your origin server. If you specify **HTTP Only**, CloudFront only uses HTTP to access the origin. If you specify **Match Viewer**, CloudFront fetches objects from your origin using HTTP or HTTPS, depending on the protocol of the viewer request.
- **HTTP Port:** The HTTP port that the custom origin listens on. Valid values include ports 80, 443, and 1024 to 65535. The default value is port 80.
- **HTTPS Port:** The HTTPS port that the custom origin listens on. Valid values include ports 80, 443, and 1024 to 65535. The default value is port 443.

Then click **Continue** and go to Step 7.

**Create Origin** [Cancel X]

CHOOSE AN ORIGIN | CREATE DEFAULT BEHAVIOR | DISTRIBUTION DETAILS | REVIEW

An origin is the Amazon S3 bucket or the web server where you store the original versions of your web content. After you create your distribution, you can add more origins in the Distribution Details pane. [Learn More](#)

**Origin Domain Name:**

**Origin ID:**

**Origin Protocol Policy:** ☒ HTTP Only ☐ Match Viewer

**HTTP Port:**

**HTTPS Port:**

< Back Continue ▶

7. On the **Create Default Cache Behavior** page, specify the following information:

- **Viewer Protocol Policy:** The protocol policy that you want viewers to use to access your content. If you specify **HTTP and HTTPS**, viewers can use both protocols. If you specify **HTTPS Only**, viewers are only allowed to access your content if they're using HTTPS.
- **Object Caching and Minimum TTL:** If your origin server is adding a `Cache-Control` header to your objects to control how long they stay in the CloudFront cache, select **Use Origin Cache Headers**.

To specify a minimum time that your objects stay in the CloudFront cache regardless of `Cache-Control` headers, select **Customize**. Then specify the minimum number of seconds that you want objects to stay in the CloudFront cache even if `Cache-Control` headers specify a lower value.

For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Object Expiration\)](#) (p. 65).

- **Forward Query Strings:** If your origin server returns different versions of an object (for example, `images/image.jpg`) based on a query string in the URL, select **Yes**.

If your origin returns the same version of an object regardless of the query string, select **No**. This increases the likelihood that CloudFront can serve a request from the cache, which improves performance and reduces the load on your origin. For more information, see [How CloudFront Forwards and Logs Query String Parameters](#) (p. 62).

Then click **Continue**.

**Create Cache Behavior** Cancel

CHOOSE AN ORIGIN CREATE DEFAULT BEHAVIOR DISTRIBUTION DETAILS REVIEW

A cache behavior determines how CloudFront communicates with your origin. After you create your distribution, you can add more cache behaviors in the Distribution Details pane. [Learn More](#)

**Path Pattern:** Default (\*)

**Origin:** S3-myawsbucket100

**Viewer Protocol Policy:** ☒ HTTP and HTTPS ☐ HTTPS Only

**Object Caching:** ☒ Use Origin Cache Headers ☐ Customize

**Minimum TTL:**  ?

**Forward Query Strings:** ☐ Yes ☒ No (Improves Caching)

[< Back](#) [Continue >](#)

8. On the **Distribution Details** page, enter the applicable values, then click **Continue**:

- **Alternate Domain Names (CNAMEs):** Optional. Specify one or more domain names that you want to use for URLs for your objects instead of the domain name that CloudFront assigns when you create your distribution. For example, if you want the URL for the object:

`images/image.jpg`

to look like this:

`http://www.example.com/images/image.jpg`

instead of like this:

`http://d111111abcdef8.cloudfront.net/images/image.jpg`

you would create a CNAME for `www.example.com`. You can create up to 10 CNAMEs per distribution.

### Note

If you add a CNAME for `www.example.com` to your distribution, you also need to create (or update) a CNAME record with your DNS service to route queries for `www.example.com` to `d111111abcdef8.cloudfront.net`. For more information, see [Using Alternate Domain Names \(CNAMEs\)](#) (p. 53).

### Important

You must own any domain name that you specify as an alternate domain name.

- **Default Root Object:** Optional. The object that you want CloudFront to request from your origin (for example, `index.html`) when a viewer requests the root URL of your distribution (`http://www.example.com/`) instead of an object in your distribution (`http://www.example.com/product-description.html`). Specifying a default root object avoids exposing the contents of your distribution.
- **Logging:** Optional. If you want CloudFront to log information about each request for an object and store the log files in an Amazon S3 bucket, select **On**, and specify the bucket and an optional prefix for the names of the log files. There is no extra charge to enable logging, but you accrue the usual

Amazon S3 charges for storing and accessing the files. CloudFront doesn't delete the logs automatically, but you can delete them at any time.

- **Comments:** Optional. Enter any comments that you want to save with the distribution.
- **Distribution State:** Select **Enabled** if you want CloudFront to begin processing requests as soon as the distribution is created, or select **Disabled** if you do not want CloudFront to begin processing requests after the distribution is created.

The screenshot shows the 'Create Distribution' wizard in the AWS Management Console, specifically the 'Distribution Details' step. The wizard has four steps: 'CHOOSE AN ORIGIN', 'CREATE DEFAULT BEHAVIOR', 'DISTRIBUTION DETAILS' (which is the current step), and 'REVIEW'. The 'DISTRIBUTION DETAILS' step includes the following fields and options:

- Alternate Domain Names (CNAMEs):** A text input field with a placeholder and a help icon. Below it, a message states 'Additional configuration is required.' with a help icon.
- Default Root Object:** A text input field.
- Logging:** Radio buttons for 'On' and 'Off'. The 'Off' option is selected.
- Bucket for Logs:** A text input field.
- Log Prefix:** A text input field.
- Comments:** A text input field with a placeholder and a help icon.
- Distribution State:** Radio buttons for 'Enabled' and 'Disabled'. The 'Enabled' option is selected.

At the bottom of the wizard, there is a '< Back' button and a 'Continue >' button with a right arrow icon.

9. On the **Review** page, review your settings, and click **Create Distribution**.
10. When CloudFront has finished creating your distribution, the value of the **Status** column for your distribution will change from **InProgress** to **Deployed**. If you chose to enable the distribution in Step 8, it will then be ready to process requests. This should take less than 15 minutes.

The domain name that CloudFront assigns to your distribution appears in the list of distributions. (It also appears on the **General** tab for a selected distribution.)

When your distribution is deployed, confirm that you can access your content using your new CloudFront URL or CNAME. For more information, see [Testing Your Download Distribution \(p. 26\)](#).

## Testing Your Download Distribution

After you've created your distribution, CloudFront knows where your origin server is, and you know the domain name associated with the distribution. You can create links to your objects using the CloudFront domain name, and CloudFront will serve the objects to your web page or application.

### Note

You must wait until the status of the distribution changes to **Deployed** before you can test your links.

### To create links to objects in a download distribution

1. Copy the following HTML code into a new file, replace *domain-name* with your distribution's domain name, and replace *object-name* with the name of your object.

```
<html>
<head>My CloudFront Test</head>
<body>
<p>My text content goes here.</p>
<p>
</html>
```

For example, if your domain name were `d111111abcdef8.cloudfront.net` and your object were `image.jpg`, the URL for the link would be:

```
http://d111111abcdef8.cloudfront.net/image.jpg.
```

If your object is in a folder on your origin server, then the folder must also be included in the URL. For example, if `image.jpg` were located in the `images` folder on your origin server, then the URL would be:

```
http://d111111abcdef8.cloudfront.net/images/image.jpg
```

2. Save the HTML code in a file that has a `.html` filename extension.
3. Open your web page in a browser to ensure that you can see your object.

The browser returns your page with the embedded image file, served from the edge location that CloudFront determined was appropriate to serve the object.

# Creating Streaming Distributions

---

The following task list describes the process for configuring on-demand streaming. If you're using JW Player, Flowplayer, or Adobe Flash Builder for your media player, see the applicable tutorial instead:

- [JW Player](#)
- [Flowplayer](#)
- [Adobe Flash Builder](#)

## Process for Streaming Media Files

1	Create an Amazon S3 bucket for your media files. If you are using a different Amazon S3 bucket for your media player, create an Amazon S3 bucket for the media player files, too.
2	Choose and configure a media player to play your media files. For more information, refer to the documentation for the media player.
3	Upload the files for your media player to the origin from which you want CloudFront to get the files. If you are using an Amazon S3 bucket as the origin for the media player, make the files (not the bucket) publicly readable.
4	Create a download distribution for your media player. (You can also use an existing distribution.) For more information, see <a href="#">Creating Download Distributions (p. 21)</a> .
5	<p>Upload your media files to the Amazon S3 bucket that you created for the media files, and make the content (not the bucket) publicly readable.</p> <p><b>Important</b></p> <p>Media files in a Flash Video container must include the .flv filename extension, or the media will not stream.</p> <p>You can put media player files and media files in the same bucket.</p>
6	<p>Create a streaming distribution for your media files:</p> <ul style="list-style-type: none"><li>• For more information about creating a download distribution using the CloudFront console, see <a href="#">Creating a Streaming Distribution Using the CloudFront Console (p. 29)</a>.</li><li>• For information about creating a download distribution using the CloudFront API, go to <a href="#">POST Streaming Distribution</a> in the <i>Amazon CloudFront API Reference</i>.</li></ul>



7	Configure your media player. For more information, see <a href="#">Configuring the Media Player (p. 51)</a> .
---	---

If you have trouble getting your content to play, see [Troubleshooting Streaming Distributions \(p. 52\)](#).

## Creating a Streaming Distribution Using the CloudFront Console

### Note

A few CloudFront settings are currently not available in the CloudFront console:

- Specifying the AWS accounts that are allowed to sign URLs for private content. For more information, see [Requiring Signed URLs \(p. 95\)](#).
- Creating an origin access identity, which prevents users from accessing files in an Amazon S3 bucket by using the Amazon S3 URL. For more information, see [Securing Your Content in Amazon S3 \(p. 90\)](#).
- Associating an origin access identity with your CloudFront distribution. For more information, see [Creating a Private Content Distribution \(p. 92\)](#).

If you want to use these settings, you must create the distribution using the CloudFront API. For information about creating a streaming distribution using the CloudFront API, go to [POST Streaming Distribution](#) in the *Amazon CloudFront API Reference*.

### To create a streaming distribution using the CloudFront console

1. Sign in to the AWS Management Console and open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Click **Create Distribution**.
3. In the **Create Distribution Wizard**, click **Streaming**, and click **Continue**.
4. Specify the distribution details:

## Amazon CloudFront Developer Guide

### Creating a Streaming Distribution Using the CloudFront Console

**Create Distribution** [Cancel]

**DISTRIBUTION DETAILS** REVIEW

**Origin Domain Name:** [Text Input]

**Alternate Domain Names (CNAMEs)** [Text Area]  
Additional configuration is required. [Help Icon]

**Logging:** ☐ On ☒ Off

**Bucket for Logs:** [Text Input]

**Log Prefix:** [Text Input]

**Comments:** [Text Area]

**Distribution State:** ☒ Enabled ☐ Disabled

< Back [Continue]

- **Origin Domain Name:** If you're using a bucket that is associated with the AWS account that you're logged on with, click in the **Origin Domain Name** field, and select the bucket from the list.

If you're using a bucket that is associated with another AWS account, enter the bucket name in the following format:

*bucket-name*.s3.amazonaws.com

- **Alternate Domain Names (CNAMEs):** Optional. Specify one or more domain names that you want to use for URLs for your objects instead of the domain name that CloudFront assigns when you create your distribution. For example, if you want the URL for the object:

images/image.jpg

to look like this:

http://www.example.com/images/image.jpg

instead of like this:

http://d111111abcdef8.cloudfront.net/images/image.jpg

you would create a CNAME for `www.example.com`. You can create up to 10 CNAMEs per distribution.

#### Note

If you add a CNAME for `www.example.com` to your distribution, you also need to create (or update) a CNAME record with your DNS service to route queries for `www.example.com` to `d111111abcdef8.cloudfront.net`. For more information, see [Using Alternate Domain Names \(CNAMEs\)](#) (p. 53).

#### Important

You must own any domain name that you specify as an alternate domain name.

- **Logging:** Optional. If you want CloudFront to log information about each request for an object and store the log files in an Amazon S3 bucket, select **On**, and specify the bucket and an optional prefix for the names of the log files. There is no extra charge to enable logging, but you accrue the usual Amazon S3 charges for storing and accessing the files. CloudFront doesn't delete the logs automatically, but you can delete them at any time. For more information, see [Access Logs \(p. 129\)](#).
- **Comments:** Optional. Enter any comments that you want to save with the distribution.
- **Distribution State:** Select **Enabled** if you want CloudFront to begin processing requests as soon as the distribution is created, or select **Disabled** if you want to wait until later.

5. Click **Continue**.
6. On the **Review** page, review your settings, and click **Create Distribution**.
7. When your distribution has been created, the value of the **Status** column for your distribution will change from **InProgress** to **Deployed**. If you chose to enable the distribution in Step 4, it will then be ready to process requests. This should take less than 15 minutes.

The domain name that CloudFront assigns to your distribution appears in the list of distributions. The domain name also appears on the **General** tab for a selected distribution.

# Migrating from Amazon S3 to CloudFront

---

If you currently distribute content from your Amazon S3 bucket using a CNAME, you can migrate to CloudFront with no disruption by using the following process.

## Process for Migrating from Amazon S3 to CloudFront

1	<p>Create a CloudFront distribution using the process described in the applicable topic:</p> <ul style="list-style-type: none"><li>• <a href="#">Creating Download Distributions</a> (p. 21)</li><li>• <a href="#">Creating Streaming Distributions</a> (p. 28)</li></ul> <p>When you create the distribution, specify the name of your Amazon S3 bucket as the origin server. If you're using a CNAME with Amazon S3, specify the CNAME for your distribution, too.</p>
2	<p>Create test links to publicly readable objects in your Amazon S3 bucket, and test the links. Use the domain name for the distribution that DNS name in the links, for example, <code>http://d1111111abcdef8.cloudfront.net/images/image.jpg</code>.</p> <p>For more information about format of CloudFront URLs, see <a href="#">Format of URLs for CloudFront Objects</a> (p. 60).</p>

3	<p>If you're using Amazon S3 CNAMEs, your application uses your domain name (for example, example.com) to reference the objects in your Amazon S3 bucket instead of using the name of your bucket (for example, myawsbucket.s3.amazonaws.com). To continue using your domain name to reference objects instead of using the CloudFront domain name for your distribution (for example, d1111111abcdef8.cloudfront.net), you need to update your settings with your DNS service provider.</p> <p>For Amazon S3 CNAMEs to work, your DNS service provider has a CNAME resource record set for your domain that currently routes end-user queries for the domain to your Amazon S3 bucket. When someone requests the object:</p> <pre>http://example.com/images/image.jpg</pre> <p>the request is automatically rerouted, and the object they see is:</p> <pre>http://myawsbucket.s3.amazonaws.com/images/image.jpg</pre> <p>To route queries to your CloudFront distribution instead of your Amazon S3 bucket, you need to use the method provided by your DNS service provider to update the CNAME resource record set for your domain. This updated CNAME record will start to redirect DNS queries from your domain to the CloudFront domain name for your distribution. For more information, see the documentation provided by your DNS service provider.</p> <p><b>Note</b></p> <p>If you're using Route 53 as your DNS service, go to <a href="#">Creating, Changing, and Deleting Resource Record Sets</a> for information about how to update a CNAME resource record set.</p> <p>For more information about using CNAMEs with CloudFront, see <a href="#">Using Alternate Domain Names (CNAMEs)</a> (p. 53).</p> <p>After you update the CNAME resource record set, it can take up to 72 hours for the change to propagate throughout the DNS system, although it usually happens faster. During this time, some requests for your content will continue to be routed to your Amazon S3 bucket, and others will be routed to CloudFront.</p>
---	--

# Working with Distributions

---

## Topics

- [Changes to the CloudFront API \(p. 34\)](#)
- [Overview of Download and Streaming Distributions \(p. 35\)](#)
- [Actions on Distributions \(p. 35\)](#)
- [Working with Download Distributions \(p. 36\)](#)
- [Working with Streaming Distributions \(p. 45\)](#)
- [Using Alternate Domain Names \(CNAMEs\) \(p. 53\)](#)
- [Listing, Viewing, and Updating CloudFront Distributions \(p. 55\)](#)
- [Deleting a Distribution \(p. 59\)](#)

A CloudFront distribution identifies the files that you want CloudFront to distribute through its edge locations and controls a variety of details about how you want CloudFront to distribute the files. When you create a distribution, you specify configuration settings such as:

- Where you want CloudFront to get the files that it distributes to edge locations—an Amazon S3 bucket or a web server.
- Whether you want the files to be available to everyone or you want to restrict access to selected users.
- Whether you want CloudFront to create access logs.

## Changes to the CloudFront API

Beginning with the 2012-05-05 version of the CloudFront API, we made substantial changes to the format of the XML document that you include in the request body when you create or update a download distribution or a streaming distribution, and when you invalidate objects. With previous versions of the API, we discovered that it was too easy to accidentally delete one or more values for an element that accepts multiple values, for example, CNAMEs and trusted signers. Our changes for the 2012-05-05 release are intended to prevent these accidental deletions and to notify you when there's a mismatch between the number of values you say you're specifying in the `Quantity` element and the number of values you're actually specifying.

Note the following about using the 2012-05-05 API version or later with download and streaming distributions that were created using earlier API versions:

- You cannot use versions of the API earlier than 2012-05-05 to update a download distribution that was created or updated using the 2012-05-05 or later CloudFront API.
- You can use the new API version to get a list of distributions, get information about a distribution, or get distribution configuration. CloudFront returns an XML document in the new XML format.
- To update a distribution that was created using an earlier API version, use the 2012-05-05 or later version of GET Distribution or GET Streaming Distribution to get an XML document in the new XML format, change the data as applicable, and use the 2012-05-05 or later version of PUT Distribution Config or PUT Streaming Distribution Config to submit the changes to CloudFront.
- You can use the new API to delete a distribution that was created using an earlier API version. The distribution must already be disabled.

## Overview of Download and Streaming Distributions

**Download distributions** serve both static and dynamic content, for example, .html, .css, .php, and image files, using HTTP or HTTPS. A download distribution gets files from locations—known as origins—that you specify. Each origin is either an Amazon S3 bucket or an HTTP server, for example, a web server. You can specify any combination of up to 10 Amazon S3 buckets and/or HTTP servers as your origins.

For more information about how download distributions work, including the values that you specify when you create a download distribution, see [Working with Download Distributions \(p. 36\)](#). For information about creating a download distribution, see [Creating Download Distributions \(p. 21\)](#).

**Streaming distributions** stream media files using the Adobe Real-Time Messaging Protocol (RTMP). A streaming distribution must use an Amazon S3 bucket as the origin.

For information about the values you specify when you create a streaming distribution, see [Working with Streaming Distributions \(p. 45\)](#). For information about creating a streaming distribution, see [Creating Streaming Distributions \(p. 28\)](#).

### Note

You can create up to 100 download distributions and 100 streaming distributions for each AWS account. The number of files that you can serve per distribution is unlimited.

## Actions on Distributions

The following table lists the actions you can perform on a distribution and provides links to the corresponding documentation on how to perform the actions using the CloudFront console and the CloudFront API.

Action	Using the CloudFront Console	Using the CloudFront API: Download Distributions	Using the CloudFront API: Streaming Distributions
Create a distribution	<b>Download Distributions:</b> See <a href="#">Creating Download Distributions</a> (p. 21) <b>Streaming Distributions:</b> See <a href="#">Creating Streaming Distributions</a> (p. 28)	Go to <a href="#">POST Distribution</a>	Go to <a href="#">POST Streaming Distribution</a>
List your distributions	See <a href="#">Listing, Viewing, and Updating CloudFront Distributions</a> (p. 55)	Go to <a href="#">GET Distribution List</a>	Go to <a href="#">GET Streaming Distribution List</a>
Get all information about a distribution	See <a href="#">Listing, Viewing, and Updating CloudFront Distributions</a> (p. 55)	Go to <a href="#">GET Distribution</a>	Go to <a href="#">GET Streaming Distribution</a>
Get the distribution configuration	See <a href="#">Listing, Viewing, and Updating CloudFront Distributions</a> (p. 55)	Go to <a href="#">GET Distribution Config</a>	Go to <a href="#">GET Streaming Distribution Config</a>
Update a distribution	See <a href="#">Listing, Viewing, and Updating CloudFront Distributions</a> (p. 55)	Go to <a href="#">PUT Distribution Config</a>	Go to <a href="#">PUT Streaming Distribution Config</a>
Delete a distribution	See <a href="#">Deleting a Distribution</a> (p. 59)	Go to <a href="#">DELETE Distribution</a>	Go to <a href="#">DELETE Streaming Distribution</a>

## Working with Download Distributions

### Topics

- [Using Amazon S3 Origins and Custom Origins for Download Distributions](#) (p. 37)
- [Values that You Specify When You Create or Update a Download Distribution Using the CloudFront Console](#) (p. 38)
- [Values that CloudFront Displays in the Console When You Create or Update a Download Distribution](#) (p. 44)
- [Requirements and Recommendations for Using Amazon EC2 and Other Custom Origins](#) (p. 45)

This section describes how you configure and manage CloudFront download distributions. For a basic explanation of distributions, see [Working with Distributions](#) (p. 34). For information about CloudFront streaming distributions, see [Working with Streaming Distributions](#) (p. 45).



## Using Amazon S3 Origins and Custom Origins for Download Distributions

When you create a download distribution, you specify where CloudFront sends requests for the files that it distributes to edge locations. CloudFront supports using Amazon S3 buckets and HTTP servers (for example, web servers) as origins.

### Using Amazon S3 Buckets for Your Origin

When you use Amazon S3 as an origin for your distribution, you place any objects that you want CloudFront to deliver in an Amazon S3 bucket. You can use any method that is supported by Amazon S3 to get your objects into Amazon S3, for example, the Amazon S3 console or API, or a third-party tool. You can create a hierarchy in your bucket to store the objects, just as you would with any other Amazon S3 bucket. You incur regular Amazon S3 charges for storing the objects in the bucket. For more information about the charges to use CloudFront, see [Paying for CloudFront \(p. 8\)](#).

#### Note

Using an existing Amazon S3 bucket as your CloudFront origin server doesn't change the bucket in any way; you can still use it as you normally would to store and access Amazon S3 objects (at the normal Amazon S3 prices).

When you specify the Amazon S3 bucket that you want CloudFront to get objects from, you use the following format:

`bucket-name.s3.amazonaws.com`

Do not specify the bucket using the following formats:

- The Amazon S3 path style, `s3.amazonaws.com/bucket-name`
- The Amazon S3 CNAME, if any

#### Important

For your bucket to work with CloudFront, the name must conform to DNS naming requirements. For more information, go to [Bucket Restrictions and Limitations](#) in the *Amazon Simple Storage Service Developer Guide*.

### Using Amazon EC2 or Other Custom Origins

A custom origin is an HTTP server, for example, a web server. The HTTP server can be an Amazon EC2 instance or an HTTP server that you manage privately. When you use a custom origin, you specify the DNS name of the server, along with the HTTP and HTTPS ports and the protocol that you want CloudFront to use when fetching objects from your origin.

Most CloudFront features are supported when you use a custom origin with the following exceptions:

- **Streaming distributions**—Not supported.
- **Private content**—Although you can use a signed URL to distribute content from a custom origin, for CloudFront to access the custom origin, the origin must remain publicly accessible. For more information, see [Using a Signed URL to Serve Private Content \(p. 84\)](#).

For information about requirements and recommendations when using custom origins, see [Requirements and Recommendations for Using Amazon EC2 and Other Custom Origins \(p. 45\)](#).

# Values that You Specify When You Create or Update a Download Distribution Using the CloudFront Console

When you create a new download distribution or update an existing distribution by using the CloudFront console, you specify the following values.

## Note

A few CloudFront settings are currently not available when you create or update a download distribution using the CloudFront console:

- Specifying trusted signers, the AWS accounts that are allowed to sign URLs for private content. For more information, see [Requiring Signed URLs \(p. 95\)](#).
- Creating an origin access identity, which prevents users from accessing files in an Amazon S3 bucket by using the Amazon S3 URL. For more information, see [Securing Your Content in Amazon S3 \(p. 90\)](#).
- Associating an origin access identity with your CloudFront distribution. For more information, see [Creating a Private Content Distribution \(p. 92\)](#).

For these settings, you must use the CloudFront API. For information about the API, see the [Amazon CloudFront API Reference](#).

General Information (p. 38)	Origins (p. 40)	Cache Behaviors (p. 41)
<a href="#">Delivery Method (p. 38)</a> <a href="#">Alternate Domain Names (CNAMEs) (p. 39)</a> <a href="#">Default Root Object (p. 39)</a> <a href="#">Logging On/Off (p. 39)</a> <a href="#">Comments (p. 40)</a> <a href="#">Amazon S3 Bucket for Log Files (p. 39)</a> <a href="#">Prefix for Log File Names (p. 40)</a> <a href="#">Distribution State (p. 40)</a>	<a href="#">Origin Domain Name (p. 40)</a> <a href="#">Origin ID (p. 41)</a> <a href="#">HTTP Port (Amazon EC2 and Other Custom Origins Only) (p. 41)</a> <a href="#">HTTPS Port (Amazon EC2 and Other Custom Origins Only) (p. 41)</a> <a href="#">Origin Protocol Policy (Amazon EC2 and Other Custom Origins Only) (p. 41)</a>	<a href="#">Path Pattern (p. 42)</a> <a href="#">Origin (p. 43)</a> <a href="#">Viewer Protocol Policy (p. 43)</a> <a href="#">Object Caching (p. 43)</a> <a href="#">Minimum TTL (p. 43)</a> <a href="#">Forward Query Strings (p. 44)</a>

For information about creating a download distribution using the CloudFront API, go to [POST Distribution](#) in the *Amazon CloudFront API Reference*. For more information about updating a download distribution using the CloudFront API, go to [PUT Distribution Config](#), also in the *Amazon CloudFront API Reference*.

## General Information

The following values apply to the entire distribution.

### Delivery Method

You specify the delivery method when you create a distribution. For a download distribution, this value is always **Download**. You can't change the delivery method for an existing distribution.

## Alternate Domain Names (CNAMEs)

Optional. One or more domain names that you want to use for URLs for your objects instead of the domain name that CloudFront assigns when you create your distribution. For example, if you want the URL for the object:

```
images/image.jpg
```

to look like this:

```
http://www.example.com/images/image.jpg
```

instead of like this:

```
http://d111111abcdef8.cloudfront.net/images/image.jpg
```

you would create a CNAME for `www.example.com`. You can create up to 10 CNAMEs per distribution.

### Note

If you add a CNAME for `www.example.com` to your distribution, you also need to create (or update) a CNAME record with your DNS service to route queries for `www.example.com` to `d111111abcdef8.cloudfront.net`.

For more information about CloudFront URLs, see [Format of URLs for CloudFront Objects \(p. 60\)](#). For more information about alternate domain names, see [Using Alternate Domain Names \(CNAMEs\) \(p. 53\)](#).

## Default Root Object

Optional. The object that you want CloudFront to request from your origin (for example, `index.html`) when a viewer requests the root URL of your distribution (`http://www.example.com/`) instead of an object in your distribution (`http://www.example.com/product-description.html`). Specifying a default root object avoids exposing the contents of your distribution.

The maximum length of the name is 255 characters. The name can contain any of the following characters:

- A-Z, a-z
- 0-9
- `_ - . * $ / ~ ' "`
- `&`, passed and returned as `&amp;`

For more information, see [Specifying a Default Root Object \(Download Distributions Only\) \(p. 72\)](#).

## Logging On/Off

Whether you want CloudFront to log information about each request for an object and store the log files in an Amazon S3 bucket. You can enable or disable logging at any time. There is no extra charge if you enable logging, but you accrue the usual Amazon S3 charges for storing and accessing the files in an Amazon S3 bucket. You can delete the logs at any time. For more information about CloudFront access logs, see [Access Logs \(p. 129\)](#).

## Amazon S3 Bucket for Log Files

Optional. The Amazon S3 bucket that you want CloudFront to store access logs in, for example, `myawslogbucket.s3.amazonaws.com`.

## Prefix for Log File Names

Optional. The string that you want CloudFront to prefix to the access log filenames for this distribution, for example, `exampleprefix/`. The slash is optional but recommended to simplify browsing your log files.

## Comments

Optional comments about the distribution. You can update the comments at any time. The maximum length is 128 characters.

## Distribution State

Indicates whether you want the distribution to be enabled or disabled once it's deployed:

- *Enabled* means that as soon as the distribution is fully deployed you can deploy links that use the distribution's domain name and end users can retrieve content. Whenever a distribution is enabled, CloudFront accepts and handles any end-user requests for content that use the domain name associated with that distribution. For more information about full deployment, see [Eventual Consistency \(p. 6\)](#).
- *Disabled* means that even though the distribution might be deployed and ready to use, end users can't use it. Whenever a distribution is disabled, CloudFront doesn't accept any end-user requests that use the domain name associated with that distribution. Until you switch the distribution from disabled to enabled (by updating the distribution's configuration), no one can use it.

You can toggle a distribution between disabled and enabled as often as you want. Follow the process for updating a distribution's configuration. For more information, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 55\)](#).

## Origins

When you create or update a distribution, you provide information about one or more locations—known as origins—where you store the original versions of your web content. CloudFront gets your web content from your origins and serves it to viewers via a world-wide network of edge servers. Each origin is either an Amazon S3 bucket or an HTTP server, for example, a web server. You can create up to 10 origins.

If you want to delete an origin, you must first edit or delete the cache behaviors that are associated with that origin.

### Caution

If you delete an origin, confirm that files that were previously served by that origin are available in another origin and that your cache behaviors are now routing requests for those files to the new origin.

When you create or update a distribution, you specify the following values for each origin.

## Origin Domain Name

The DNS domain name of the Amazon S3 bucket or HTTP server from which you want CloudFront to get objects for this origin, for example, `myawsbucket.s3.amazonaws.com` or `www.example.com`. In the CloudFront console, the **Origin Domain Name** list enumerates the AWS resources that are associated with the current AWS account. To use a resource from a different AWS account or to use a non-AWS resource, type the domain name of the resource. For example, for an Amazon S3 bucket, type the name in the format `bucketname.s3.amazonaws.com`.

If the origin is an HTTP server, the files must be publicly readable. If the origin is an Amazon S3 bucket, the files must be publicly readable unless you secure your content in Amazon S3 by using a CloudFront origin access identity. For more information, see [Securing Your Content in Amazon S3 \(p. 90\)](#).

## Origin ID

A string that uniquely distinguishes this origin from other origins in this distribution. When you create a cache behavior, you use the origin ID that you specify here to identify the origin to which you want CloudFront to route requests when the requests match the path pattern for that cache behavior. For more information, see [Cache Behaviors \(p. 41\)](#).

## Origin Protocol Policy (Amazon EC2 and Other Custom Origins Only)

The protocol policy that you want CloudFront to use when fetching objects from your origin server. If you specify **HTTP Only**, CloudFront only uses HTTP to access the origin. If you specify **Match Viewer**, CloudFront fetches objects from your origin using HTTP or HTTPS, depending on the protocol of the viewer request.

## HTTP Port (Amazon EC2 and Other Custom Origins Only)

Optional. The HTTP port that the custom origin listens on. Valid values include ports 80, 443, and 1024 to 65535. The default value is port 80.

## HTTPS Port (Amazon EC2 and Other Custom Origins Only)

Optional. The HTTPS port that the custom origin listens on. Valid values include ports 80, 443, and 1024 to 65535. The default value is port 443.

## Cache Behaviors

A cache behavior lets you configure a variety of CloudFront functionality for a given URL path pattern for files on your website. For example, one cache behavior might apply to all `.jpg` files in the `images` directory on a web server that you're using as an origin server for CloudFront. The functionality you can configure for each cache behavior includes:

- The path pattern.
- If you have configured multiple origins for your CloudFront distribution, which origin you want CloudFront to forward your requests to.
- Whether to forward query strings to your origin.
- Whether accessing the specified files requires signed URLs.
- Whether to require end users to use HTTPS to access those files.
- The minimum amount of time that those files stay in the CloudFront cache regardless of the value of any `Cache-Control` headers that your origin adds to the files.

When you create a new distribution, you specify settings for the default cache behavior, which automatically forwards all requests to the origin that you specify when you create the distribution. After you create a distribution, you can create additional cache behaviors that define how CloudFront responds when it receives a request for objects that match a path pattern, for example, `*.jpg`. If you create additional cache behaviors, the default cache behavior is always the last to be processed. Other cache behaviors are processed in the order in which they're listed in the CloudFront console or, if you're using the CloudFront API, the order in which they're listed in the `DistributionConfig` element for the distribution. For more information, see [Path Pattern \(p. 42\)](#).

When you create a cache behavior, you specify the one origin from which you want CloudFront to get objects. As a result, if you want CloudFront to distribute objects from all of your origins, you must have at least as many cache behaviors (including the default cache behavior) as you have origins. For example,

if you have two origins and only the default cache behavior, the default cache behavior will cause CloudFront to get objects from one of the origins, but the other origin will never be used.

## Path Pattern

The pattern (for example, `images/*.jpg`) that specifies which requests you want this cache behavior to apply to. When CloudFront receives an end-user request, the requested path is compared with path patterns in the order in which cache behaviors are listed in the distribution. The first match determines which cache behavior is applied to that request. For example, suppose you have three cache behaviors with the following three path patterns, in this order:

- `images/*.jpg`
- `images/*`
- `*.gif`

A request for the file `images/sample.gif` doesn't satisfy the first path pattern, so the associated cache behaviors are not be applied to the request. The file does satisfy the second path pattern, so the cache behaviors associated with the second path pattern are applied even though the request also matches the third path pattern.

### Caution

Define path patterns and their sequence carefully or you may give end users undesired access to your content. For example, suppose a request matches the path pattern for two cache behaviors. The first cache behavior does not require signed URLs and the second cache behavior does require signed URLs. End users will be able to access the objects without using a signed URL because CloudFront processes the cache behavior associated with the first match.

The path you specify applies to requests for all files in the specified directory and in subdirectories below the specified directory. For example, if an `images` directory contains `product1` and `product2` subdirectories, the path pattern `images/*.jpg` applies to requests for any `.jpg` file in the `images`, `images/product1`, and `images/product2` directories. If you want to apply a different cache behavior to the files in the `images/product1` directory than the files in the `images` and `images/product2` directories, create a separate cache behavior for `images/product1` and move that cache behavior to a position above (before) the cache behavior for the `images` directory.

You can use the following wildcard characters in your path pattern:

- `*` matches 0 or more characters.
- `?` matches exactly 1 character.

The following examples show how the wildcard characters work:

Path pattern	Files that match the path pattern
<code>*.jpg</code>	All <code>.jpg</code> files
<code>images/*.jpg</code>	All <code>.jpg</code> files in the <code>images</code> directory and in subdirectories under the <code>images</code> directory
<code>a*.jpg</code>	<ul style="list-style-type: none"><li>• All <code>.jpg</code> files for which the filename begins with <code>a</code>, for example, <code>apple.jpg</code> and <code>appalachian_trail_2012_05_21.jpg</code></li><li>• All <code>.jpg</code> files for which the file path begins with <code>a</code>, for example, <code>abra/cadabra/magic.jpg</code>.</li></ul>

Path pattern	Files that match the path pattern
a?? .jpg	All .jpg files for which the filename begins with a and is followed by exactly two other characters, for example, ant .jpg and abe .jpg
*.doc*	All files for which the filename extension begins with .doc, for example, .doc, .docx, and .docm files. You can't use the path pattern *.doc? in this case, because that path pattern wouldn't apply to requests for .doc files; the ? wildcard character replaces exactly one character.

The path pattern for the default cache behavior is \* (all files) and cannot be changed. If the request for an object does not match the path pattern for any of the other cache behaviors, CloudFront applies the behavior that you specify in the default cache behavior.

The maximum length of a path pattern is 255 characters. The value can contain any of the following characters:

- A-Z, a-z
- 0-9
- \_ - . \* \$ / ~ ' ' @ : +
- &, passed and returned as `&amp;`

## Origin

When you're creating a download distribution, the value that you specified for **Origin ID** on the **Create Origin** page of the **Create Distribution Wizard**.

When you're adding cache behaviors to an existing origin or updating an existing origin, the value of **Origin ID** for the origin that you want CloudFront to route requests to when a request matches the path pattern either for a cache behavior or for the default cache behavior.

## Viewer Protocol Policy

Specifies whether to allow access to your content in the origin specified by **Origin** using both HTTP and HTTPS or to restrict access to HTTPS requests. For more information, see [Using an HTTPS Connection to Access Your Objects \(p. 126\)](#).

## Object Caching

If your origin server is adding a `Cache-Control` header to your objects to control how long the objects stay in the CloudFront cache, select **Use Origin Cache Headers**.

To specify a minimum time that your objects stay in the CloudFront cache regardless of `Cache-Control` headers, select **Customize**. Then, in the **Minimum TTL** field, specify the minimum number of seconds that you want objects to stay in the CloudFront cache even if `Cache-Control` headers specify a lower value.

## Minimum TTL

The minimum amount of time that you want objects to stay in CloudFront caches before CloudFront queries your origin to see whether the object has been updated. For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Object Expiration\) \(p. 65\)](#).

## Forward Query Strings

Indicates whether you want CloudFront to forward query strings to the origin that is associated with this cache behavior. For more information about query strings, see [How CloudFront Forwards and Logs Query String Parameters](#) (p. 62).

# Values that CloudFront Displays in the Console When You Create or Update a Download Distribution

When you create a new distribution or update an existing distribution, CloudFront displays the following information in the CloudFront console.

### Note

A few CloudFront settings are currently not visible in the CloudFront console:

- A list of active trusted signers, the AWS accounts that are allowed to sign URLs for private content.
- CloudFront origin access identities, which you can use to prevent users from accessing files in an Amazon S3 bucket by using the Amazon S3 URL.
- The origin access identity that is associated with your CloudFront distribution.
- 

To see these settings, you must use the CloudFront API. For information about the API, see the [Amazon CloudFront API Reference](#).

## Distribution ID

When you perform an action on a distribution using the CloudFront API, you use the distribution ID to specify which distribution you want to perform the action on, for example, `EDFDVBD6EXAMPLE`. You can't change the distribution ID.

## Status

The possible status values for a distribution are listed in the following table.

Value	Description
<b>InProgress</b>	The distribution is still being created or updated.
<b>Deployed</b>	The distribution has been created or updated and the changes have been fully propagated through the CloudFront system.

### Note

In addition to ensuring that the status for a distribution is **Deployed**, you must enable the distribution before end users can use CloudFront to access your content. For more information, see [Distribution State](#) (p. 40).



## Last Modified

The last modification time stamp uses the ISO 8601 format, for example, 2012-05-19T19:37:58Z. For more information, go to <http://www.w3.org/TR/NOTE-datetime>.

## Domain Name

You use the distribution's domain name in the links to your objects. For example, if your distribution's domain name is `d1111111abcdef8.cloudfront.net`, the link to `images/image.jpg` would be `http://d1111111abcdef8.cloudfront.net/images/image.jpg`. You can't change the CloudFront domain name for your distribution. For more information about CloudFront URLs for links to your objects, see [Format of URLs for CloudFront Objects](#) (p. 60).

If you specified one or more alternate domain names (CNAMEs), you can use your own domain names for links to your objects instead of using the CloudFront domain name. For more information about CNAMEs, see [Alternate Domain Names \(CNAMEs\)](#) (p. 39).

### Note

CloudFront domain names are unique. Your distribution's domain name was never used for a previous distribution and will never be reused for another distribution in the future.

## Requirements and Recommendations for Using Amazon EC2 and Other Custom Origins

Follow these guidelines for using Amazon EC2 instances and other custom origins with CloudFront.

- Host and serve the same content on all servers.
- Log the `X-Amz-Cf-Id` header entries on all servers; CloudFront requires this information for debugging.
- Restrict access requests to the HTTP and HTTPS ports that your custom origin listens on.
- Synchronize the clocks of all servers in your implementation.
- Use redundant servers to handle failures.
- For information about request and response behavior and about supported HTTP status codes, see [Request and Response Behavior, and Supported HTTP Status Codes](#) (p. 77).

If you use Amazon Elastic Compute Cloud for your custom origins, we recommend that you do the following:

1. Use an Amazon Machine Image that automatically installs the software for a web server. For more information, see the [Amazon EC2 documentation](#).
2. Use an Elastic Load Balancing load balancer to handle traffic across multiple Amazon EC2 instances and to isolate your application from changes to Amazon EC2 instances. For example, if you use a load balancer, you can add and delete Amazon EC2 instances without changing your application. For more information, see the [Elastic Load Balancing documentation](#).
3. When you create your CloudFront distribution, specify the URL of the load balancer for the domain name of your origin server. For more information, see [Creating Download Distributions](#) (p. 21).

## Working with Streaming Distributions

### Topics

- [How Streaming Distributions Work](#) (p. 46)

- [Using an Amazon S3 Bucket as the Origin for a Streaming Distribution \(p. 48\)](#)
- [Creating Multiple Streaming Distributions for an Origin Server \(p. 48\)](#)
- [Values that You Specify When You Create a Streaming Distribution \(p. 49\)](#)
- [Values that CloudFront Returns When You Create a Streaming Distribution \(p. 50\)](#)
- [Configuring the Media Player \(p. 51\)](#)
- [Restricting Access Using Crossdomain.xml \(p. 52\)](#)
- [Error Codes for Streaming Distributions \(p. 52\)](#)
- [Troubleshooting Streaming Distributions \(p. 52\)](#)

This section describes how you configure and manage streaming distributions. For information about how to create a streaming distribution, see [Creating Streaming Distributions \(p. 28\)](#).

## How Streaming Distributions Work

To stream media files using CloudFront, you provide two types of files to your end users:

- Your media files
- A media player, for example, JW Player, Flowplayer, or Adobe Flash Builder

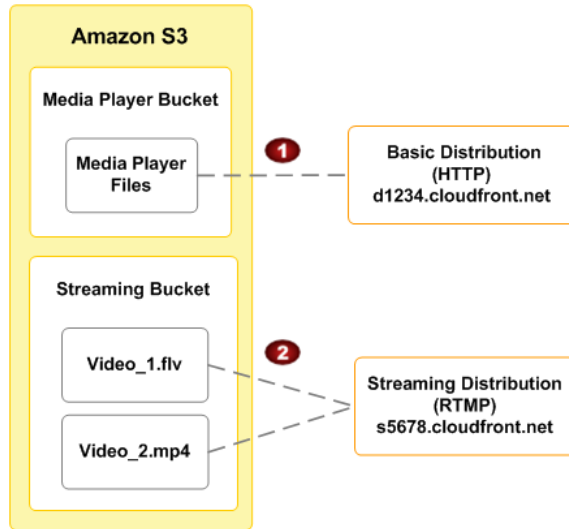
End users view your media files using the media player that you provide for them; they do not use the media player (if any) that is already installed on their computer or other device.

When an end user streams your media file, the media player begins to play the content of the file while the file is still being downloaded from CloudFront. The media file is not stored locally on the end user's system.

To use CloudFront to serve both the media player and the media files, you need two types of distributions: a download distribution for the media player, and a streaming distribution for the media files. Download distributions serve files over HTTP, while streaming distributions stream media files over RTMP (or a variant of RTMP).

The following example assumes that your media files and your media player are stored in different buckets in Amazon S3, but that isn't required—you can store media files and your media player in the same Amazon S3 bucket. You can also make the media player available to end users in other ways, for example, using CloudFront and a custom origin. However, the media files must use an Amazon S3 bucket as the origin.

In the following diagram, your site serves a cached copy of the media player to each end user through the `d1234.cloudfront.net` domain. The media player then accesses cached copies of your media files through the `s5678.cloudfront.net` domain.



<b>1</b>	Your media player bucket holds the media player and is the origin server for a regular HTTP distribution. In this example, the domain name for the distribution is <code>d1234.cloudfront.net</code> . (The <code>d</code> in <code>d1234.cloudfront.net</code> indicates that this is a download distribution.)
<b>2</b>	Your streaming media bucket holds your media files and is the origin server for a streaming distribution. In this example, the domain name for the distribution is <code>s5678.cloudfront.net</code> . (The <code>s</code> in <code>s5678.cloudfront.net</code> indicates that this is a streaming distribution.)

When you configure CloudFront to distribute media files, CloudFront uses Adobe Flash Media Server 3.5 as the streaming server and streams your media files using Adobe's Real-Time Messaging Protocol (RTMP). CloudFront accepts RTMP requests over port 1935 and port 80.

CloudFront supports the following variants of the RTMP protocol:

- **RTMP**—Adobe's Real-Time Message Protocol
- **RTMPT**—Adobe streaming tunneled over HTTP
- **RTMPE**—Adobe encrypted
- **RTMPE**—Adobe encrypted tunneled over HTTP

For a basic summary of RTMP and the file formats that Adobe Flash Media Server supports, go to [Overview of Streaming with Flash Media Server 3](#) on the Adobe website. The overview includes information about supported codecs and containers.

Resources available on the Internet can help you determine the bit rate to use for your Flash files, for example, the [Flash video \(FLV\) bitrate calculator](#) on the Adobe website.

CloudFront supports all of the features in Adobe Flash Media Server 3.5 related to *dynamic streaming*, which lets you switch between streams of different qualities during playback. For more information, go to [Dynamic streaming in Flash Media Server 3.5: Part 1](#) on the Adobe website.

To serve streamed content, you need to provide your end users with a media player:

- You can write your own player using Adobe Flash. For more information, go to <http://www.adobe.com/products/flashplayer/>.
- You can use an existing player, such as JW Player. For more information, go to <http://www.longtailvideo.com/>.

## Using an Amazon S3 Bucket as the Origin for a Streaming Distribution

When you create a distribution, you specify where CloudFront gets the files that it distributes to edge locations. For a streaming distribution, you must use an Amazon S3 bucket; custom origins are not supported. To get your objects into your bucket, you can use any method supported by Amazon S3, for example, the Amazon S3 API or a third-party tool. You can create a hierarchy in your bucket just as you would with any other Amazon S3 bucket. You incur regular Amazon S3 charges for storing the objects in the bucket. For more information about the charges to use CloudFront, see [Paying for CloudFront \(p. 8\)](#).

Using an existing Amazon S3 bucket as your CloudFront origin server doesn't change the bucket in any way; you can still use it as you normally would to store and access Amazon S3 objects (at the normal Amazon S3 prices).

You can use the same Amazon S3 bucket for both streaming and download distributions.

### Note

After you create a streaming distribution, you can't change its origin server. If you need to change the Amazon S3 bucket for a streaming distribution, you must create a new distribution that uses the new bucket and update either your links or your DNS records to use the domain name for the new distribution. You can then delete the original distribution. For more information, see [Deleting a Distribution \(p. 59\)](#).

When you specify the name of the Amazon S3 bucket that you want CloudFront to get objects from, you use the following format:

`bucket-name.s3.amazonaws.com`

Do not specify the name of the bucket using the following values:

- The Amazon S3 path style, `s3.amazonaws.com/bucket-name`
- The Amazon S3 CNAME, if any

### Important

For your bucket to work with CloudFront, the name must conform to DNS naming requirements. For more information, go to [Bucket Restrictions and Limitations](#) in the *Amazon Simple Storage Service Developer Guide*.

## Creating Multiple Streaming Distributions for an Origin Server

You typically create one streaming distribution per Amazon S3 bucket, but you can choose to create multiple streaming distributions for the same bucket. For example, if you had two distributions for an Amazon S3 bucket, you could reference a single media file using either distribution. In this case, if you had a media file called `media.flv` in your origin server, CloudFront would work with each distribution as though it referenced an individual `media.flv` object: one `media.flv` accessible through one distribution, and another `media.flv` accessible through the other distribution.

## Values that You Specify When You Create a Streaming Distribution

To stream media files using CloudFront, you create a streaming distribution and specify the following values.

### Amazon S3 Bucket

Specify the name of the Amazon S3 bucket that you want CloudFront to get objects from using the following format:

*bucket-name*.s3.amazonaws.com

For more information, see [Using an Amazon S3 Bucket as the Origin for a Streaming Distribution](#) (p. 48).

### Origin Access Identity

Optional. Use the origin access identity if you want to configure the distribution so end users can only access objects in an Amazon S3 bucket through CloudFront, not by specifying the Amazon S3 URL for the objects. For more information about origin access identity, see [Using a Signed URL to Serve Private Content](#) (p. 84).

### CallerReference (API Only)

The caller reference is a unique value that you provide and CloudFront uses to prevent replays of your request. You must provide a new caller reference value and other new information in the request for CloudFront to create a new distribution. You could use a time stamp for the caller reference, for example: 20120229090000.

If you pass the same caller reference value and the rest of the request is the same, CloudFront doesn't create a new distribution. Instead, it returns information about the distribution you previously created with that caller reference.

If you pass the same caller reference value but change other information in the request, CloudFront returns a `DistributionAlreadyExists` error. For more information about errors, see [Error Responses](#) (p. 146).

After you create a distribution, you can't change its caller reference.

### CNAME

Optional. You can associate one or more CNAME aliases with a distribution so that you can use your domain name (for example, example.com) in the URLs for your objects instead of using the domain name that CloudFront assigned when you created your distribution. For more information, see [Using Alternate Domain Names \(CNAMEs\)](#) (p. 53).

### Comment

Optional. When you create a distribution, you can include a comment of up to 128 characters about the distribution. You can update the comment at any time.

### Distribution State

When you create a distribution, you must specify whether you want the distribution to be enabled or disabled after it's created:

- *Enabled* means that as soon as the distribution is fully deployed you can deploy links that use the distribution's domain name and end users can retrieve content. Whenever a distribution is enabled, CloudFront accepts and processes any end-user requests for content that use the domain name associated with that distribution. For more information about full deployment, see [Eventual Consistency](#) (p. 6).
- *Disabled* means that even though the distribution might be deployed and ready to use, end users can't use it. When a distribution is disabled, CloudFront doesn't accept any end-user requests that use the domain name associated with that distribution. Until you switch the distribution from disabled to enabled (by updating the distribution's configuration), no one can use it.

You can toggle a distribution between disabled and enabled as often as you want. For information about updating a distribution's configuration, see [Listing, Viewing, and Updating CloudFront Distributions](#) (p. 55).

## Amazon S3 Bucket for Log Files

Optional. The Amazon S3 bucket that you want CloudFront to store the access logs in, for example, `myawslogbucket.s3.amazonaws.com`. If you enable logging, CloudFront records information about each end-user request for an object and stores the files in the specified Amazon S3 bucket. You can enable or disable logging at any time. For more information about CloudFront access logs, see [Access Logs](#) (p. 129).

## Prefix for File Names in Amazon S3 Log Bucket

Optional. The string that you want CloudFront to prefix to the access log filenames for this distribution, for example, `exampleprefix/`. For more information about CloudFront access logs, see [Access Logs](#) (p. 129).

# Values that CloudFront Returns When You Create a Streaming Distribution

When you create a new streaming distribution, CloudFront returns the following information:

## Distribution ID

When you perform an action on a distribution using the CloudFront API, you use the distribution ID to specify which distribution you want to perform the action on, for example, `EDFDVBD6EXAMPLE`.

## Status

The possible status values for a distribution are listed in the following table.

Value	Description
InProgress	The distribution is still being created or updated.
Deployed	The distribution has been created or updated and the changes have been fully propagated through the CloudFront system.

### Note

Even if the distribution's status is `Deployed`, you still must enable the distribution for use before end users can retrieve content. For more information, see [Distribution State](#) (p. 49).

## Last Modification Date

The last modification time stamp uses the ISO 8601 format, for example, 2012-05-19T19:37:58Z. For more information, go to <http://www.w3.org/TR/NOTE-datetime>.

## Domain Name

You use the distribution's domain name in the links to your objects, unless you're using alternate domain names (CNAMEs). For example, if your distribution's domain name is `d1111111abcdef8.cloudfront.net`, the link to the example `images/image.jpg` file would be `http://d1111111abcdef8.cloudfront.net/images/image.jpg`. (For more information about CNAMEs, see [Using Alternate Domain Names \(CNAMEs\)](#) (p. 53)).

### Note

CloudFront domain names are unique. Your distribution's domain name was never used for a previous distribution and will never be reused for another distribution in the future.

## Active Trusted Signers

CloudFront returns a list of the active trusted signers for the distribution if you specified active trusted signers when you created the distribution. For more information about active trusted signers and serving private content using a signed URL, see [Using a Signed URL to Serve Private Content](#) (p. 84). Active trusted signers are trusted signers who have at least one active key pair that CloudFront is aware of. Only active trusted signers can create working signed URLs.

## Configuring the Media Player

To play a media file, you must configure the media player with the correct path to the file. How you configure the media depends on which media player you're using and how you're using it.

When you configure the media player, the path you specify to the media file must contain the characters `cfx/st` immediately after the domain name, for example:

```
rtmp://s5c39gqb8ow64r.cloudfront.net/cfx/st/mediafile.flv.
```

### Note

CloudFront follows Adobe's FMS naming requirements. Different players have their own rules about how to specify streams. The example above is for JW Player. Check your player's documentation. For example, Adobe's Flash Media Server does not allow the `.flv` extension to be present on the play path. Many players remove the `.flv` extension for you.

Your media player might ask for the path separate from the file name. For example, with JW Player's wizard, you specify a `streamer` and `file` variable:

- **streamer**—`rtmp://s5c39gqb8ow64r.cloudfront.net/cfx/st` (with no trailing slash)
- **file**—`mediafile.flv`

If you've stored the media files in a directory in your bucket (for example, `videos/mediafile.flv`), then the variables for JW Player would be:

- **streamer**—`rtmp://s5c39gqb8ow64r.cloudfront.net/cfx/st` (with no trailing slash)
- **file**—`videos/mediafile.flv`

To use the JW Player wizard, go to  
<http://www.longtailvideo.com/support/jw-player-setup-wizard?example=204>.

## MPEG Files

To serve MP3 audio files or H.264/MPEG-4 video files, you might need to prefix the file name with `mp3:` or `mp4:`. Some media players can be configured to add the prefix automatically. The media player might also require you to specify the file name without the file extension (for example, `magicvideo` instead of `magicvideo.mp4`).

## Restricting Access Using Crossdomain.xml

The Adobe Flash Media Server `crossdomain.xml` file specifies which domains can access media files in a particular domain. CloudFront supplies a default file that allows all domains to access the media files in your streaming distribution, and you cannot change this behavior. If you include a more restrictive `crossdomain.xml` file in your Amazon S3 bucket, CloudFront ignores it.

## Error Codes for Streaming Distributions

The following table lists the error codes that CloudFront can send to your media player. The errors are part of the string returned with `Event.info.application.message` or `Event.info.description`.

Error	Description
<code>DistributionNotFound</code>	The distribution was not found.
<code>DistributionTypeMismatch</code>	The distribution is not a streaming distribution.
<code>InvalidInstance</code>	The instance is invalid.
<code>InvalidURI</code>	The URI is invalid.

## Troubleshooting Streaming Distributions

If you're having trouble getting your media files to play, check the following items.

Item to Check	Description
Separate distributions for the media player files and media files	The media player must be served by a regular HTTP distribution (for example, domain name <code>d11111abcdef8.cloudfront.net</code> ), and media files must be served by a streaming distribution (for example, domain name <code>s5c39gqb8ow64r.cloudfront.net</code> ). Make sure you're not using the same distribution for both.
<code>/cfx/st</code> in the file path	Confirm that the path for the file includes <code>/cfx/st</code> . You don't need to include <code>/cfx/st</code> in the path to the object in the Amazon S3 bucket. For more information, see <a href="#">Configuring the Media Player (p. 51)</a> .
MPEG-4 file names	Some media players require <code>mp4:</code> before the file name. Some might also require you to exclude the <code>.mp4</code> extension. For more information, see <a href="#">MPEG Files (p. 52)</a> .



Item to Check	Description
Port 1935 on your firewall	Adobe Flash Media Server uses port 1935 for RTMP. Make sure your firewall has this port open. If it doesn't, the typical message returned is "Unable to play video." You can also switch to RTMPT to tunnel over HTTP using port 80.
Adobe Flash Player messaging	By default, the Adobe Flash Player won't display a message if the video file it's trying to play is missing. Instead, it waits for the file to show up. You might want to change this behavior to give your end users a better experience. To instead have the player send a message if the video is missing, use <code>play("vid",0,-1)</code> instead of <code>play("vid")</code> .

## Using Alternate Domain Names (CNAMEs)

### What Is an Alternate Domain Name?

In CloudFront, an alternate domain name, also known as a CNAME, lets you use your own domain name (for example, `www.example.com`) for links to your objects instead of using the domain name that CloudFront assigns to your distribution. Both download and streaming distributions support CNAMEs.

When you create a distribution, CloudFront returns a domain name for the distribution, for example:

```
d111111abcdef8.cloudfront.net
```

If you were to use the CloudFront domain name for your objects, the URL for an object called `images/image.jpg` would be:

```
http://d111111abcdef8.cloudfront.net/images/image.jpg
```

If you want to use your own domain name, `www.example.com`, for the URLs for your objects instead of the `cloudfront.net` domain name that CloudFront assigned to your distribution, you can create a CNAME for `www.example.com`. You can then use the following URL for `images/image.jpg`:

```
http://www.example.com/images/image.jpg
```

#### Important

Note the following restrictions on using CNAMEs:

- You must have permission to create a CNAME record with the DNS service provider for the domain. Typically, this means that you own the domain, but you may also be developing an application for the domain owner.
- The DNS protocol does not allow you to create a CNAME record for the top node of a DNS namespace, also known as the zone apex. For example, if you register the DNS name `example.com`, the zone apex is `example.com`. You cannot create a CNAME record for `example.com`, but you can create CNAME records for `www.example.com`, `newproduct.example.com`, and so on.
- CloudFront doesn't support CNAMEs with HTTPS. For more information, see [Alternate Domain Names \(CNAMEs\) and HTTPS \(p. 128\)](#).

## Adding a CNAME

The following task list describes the process for using the CloudFront console to add CNAMEs to your distribution so you can use your own domain name in your links instead of the CloudFront domain name that is associated with your distribution.

You can also update your distribution using the CloudFront API:

- To update a download distribution, use the `PUT Distribution Config` API action. For more information, go to [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.
- To update a streaming distribution, use the `PUT Streaming Distribution Config` API action. For more information, go to [PUT Streaming Distribution Config](#) in the *Amazon CloudFront API Reference*.

### Process for Adding CNAMEs Using the CloudFront Console

1	Sign in to the AWS Management Console and open the Amazon CloudFront console at <a href="https://console.aws.amazon.com/cloudfront/">https://console.aws.amazon.com/cloudfront/</a> .
2	In the CloudFront console, update your distribution to include your domain name as an alternate domain name in the <b>Alternate Domain Names(CNAMEs)</b> field. For more information, see <a href="#">Listing, Viewing, and Updating CloudFront Distributions</a> (p. 55).
3	In the CloudFront console, on the General tab for your distribution, confirm that the status of your distribution has changed to <b>Deployed</b> . If you try to use an alternate domain name before the updates to your distribution have been deployed, the links you create in the following steps might not work.
4	<p>Using the method provided by your DNS service provider, add a CNAME resource record set to the hosted zone for your domain. This new CNAME resource record set will redirect DNS queries from your domain (for example, <code>www.example.com</code>) to the CloudFront domain name for your distribution (for example, <code>d111111abcdef8.cloudfront.net</code>). For more information, see the documentation provided by your DNS service provider.</p> <p><b>Note</b></p> <p>If you're using Route 53 as your DNS service, go to <a href="#">Creating, Changing, and Deleting Resource Record Sets</a> for information about how to add a CNAME resource record set.</p> <p><b>Important</b></p> <p>If you already have an existing CNAME record for your domain name, update that resource record set or replace it with a new one that points to the CloudFront domain name for your distribution.</p> <p>In addition, confirm that your CNAME resource record set points to your distribution's domain name and not to one of your origin servers.</p>

5	<p>Using dig or a similar tool, confirm that the CNAME resource record set that you created in Step 4 points to the domain name for your distribution. For more information about dig, go to <a href="http://www.kloth.net/services/dig.php">http://www.kloth.net/services/dig.php</a>.</p> <p>The following example shows a dig request on the images.example.com domain, as well as the relevant part of the response.</p> <pre>[prompt]&gt; dig images.example.com  ; &lt;&lt;&gt; DiG 9.3.3rc2 &lt;&lt;&gt; images.example.com ;; global options:  printcmd ;; Got answer: ;; -&gt;&gt;HEADER&lt;&lt;- opcode: QUERY, status: NOERROR, id: 15917 ;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 2, ADDITIONAL: 0  ;; QUESTION SECTION: ;images.example.com.      IN      A  ;; ANSWER SECTION: images.example.com. 10800 IN  CNAME  d11111abcdef8.cloudfront.net. ... ...</pre> <p>The line in the Answer Section shows a CNAME resource record set that routes queries for images.example.com to the CloudFront distribution domain name d11111abcdef8.cloudfront.net. The CNAME resource record set is configured correctly if the name on the right side of CNAME is the domain name for your CloudFront distribution. If that is any other value, for example, the domain name for your Amazon S3 bucket, then the CNAME resource record set is configured incorrectly. In that case, go back to Step 4 and correct the CNAME record to point to the domain name for your distribution.</p>
6	<p>Test the alternate domain name by creating some test links that use your domain name in the URL instead of the CloudFront domain name for your distribution.</p>
7	<p>In your application, change the links for your objects to use your alternate domain name instead of the domain name of your CloudFront distribution.</p>

## Listing, Viewing, and Updating CloudFront Distributions

You can use the CloudFront console to list the CloudFront distributions that are associated with your AWS account, view the settings for a distribution, and update most settings. Note that the following settings are currently not available in the CloudFront console:

- Specifying trusted signers, the AWS accounts that are allowed to sign URLs for private content. For more information, see [Requiring Signed URLs \(p. 95\)](#).
- Creating an origin access identity, which prevents users from accessing files in an Amazon S3 bucket by using the Amazon S3 URL. For more information, see [Securing Your Content in Amazon S3 \(p. 90\)](#).
- Associating an origin access identity with your CloudFront distribution. For more information, see [Creating a Private Content Distribution \(p. 92\)](#).

For these settings, you must use the CloudFront API. For information about which CloudFront API actions to use to list, view, and update download and streaming distributions, see [Actions on Distributions \(p. 35\)](#).

## To List, View, and Update CloudFront Distributions Using the CloudFront Console

1. Sign in to the AWS Management Console and open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In the top pane of the CloudFront console, select the distribution that you want to view or update.

### Note

The top pane lists all of the distributions that are associated with the AWS account that you used when you signed in to the CloudFront console.

3. To edit streaming distribution settings, skip to Step 6.

To edit general settings for a download distribution, perform the following steps.

- a. In the **Distribution Details** pane, on the **General** tab, click **Edit**.
  - b. Enter or update the applicable values. For more information about the fields, see [General Information \(p. 38\)](#).
- **Alternate Domain Names (CNAMEs):** Optional. Specify one or more domain names that you want to use for URLs for your objects instead of the domain name that CloudFront assigns when you create your distribution. For example, if you want the URL for the object:

`images/image.jpg`

to look like this:

`http://www.example.com/images/image.jpg`

instead of like this:

`http://d111111abcdef8.cloudfront.net/images/image.jpg`

you would create a CNAME for `www.example.com`. You can create up to 10 CNAMEs per distribution.

### Note

If you add a CNAME for `www.example.com` to your distribution, you also need to create (or update) a CNAME record with your DNS service to route queries for `www.example.com` to `d111111abcdef8.cloudfront.net`. For more information, see [Using Alternate Domain Names \(CNAMEs\) \(p. 53\)](#).

### Important

You must own any domain name that you specify as an alternate domain name.

- **Default Root Object:** Optional. The object that you want CloudFront to request from your origin (for example, `index.html`) when a viewer requests the root URL of your distribution (`http://www.example.com/`) instead of an object in your distribution (`http://www.example.com/product-description.html`). Specifying a default root object avoids exposing the contents of your distribution.
- **Logging:** Optional. If you want CloudFront to log information about each request for an object and store the log files in an Amazon S3 bucket, select **On**, and specify the bucket and an optional prefix for the names of the log files. There is no extra charge to enable logging, but you accrue the usual Amazon S3 charges for storing and accessing the files. CloudFront doesn't delete the logs automatically, but you can delete them at any time.
- **Comments:** Optional. Enter any comments that you want to save with the distribution.
- **Distribution State:** Select **Enabled** if you want CloudFront to begin processing requests as soon as the distribution is created, or select **Disabled** if you want to wait until later.

- c. Click **Yes, Edit**.
4. To add or update an origin for a download distribution, perform the following steps.
    - a. In the **Distribution Details** pane, click the **Origin** tab.
    - b. Click **Create Origin** or **Edit**.
    - c. Enter or update the applicable values. For more information about the fields, see [Origins \(p. 40\)](#).

- **Origin Domain Name:** If you're using an Amazon S3 bucket as the origin and if the bucket is associated with the AWS account that you're signed in with, click in the **Origin Domain Name** field, and select the bucket from the list.

If you're using a bucket that is associated with another AWS account, enter the bucket name in the following format:

*bucket-name*.s3.amazonaws.com

If you're using an HTTP server as the origin for your distribution, enter the domain name of your HTTP server, for example, *www.example.com*.

- **Origin ID:** A string that uniquely distinguishes this origin from other origins in this distribution. Enter a value that will help you identify each origin when you create or update cache behaviors and have to specify an origin using its origin ID.

If you're using an HTTP server as the origin for your distribution, the following fields appear after you enter a domain name.

- **Origin Protocol Policy:** The protocol policy that you want CloudFront to use when fetching objects from your origin server. If you specify **HTTP Only**, CloudFront only uses HTTP to access the origin. If you specify **Match Viewer**, CloudFront fetches objects from your origin using HTTP or HTTPS, depending on the protocol of the viewer request.
- **HTTP Port:** The HTTP port that the custom origin listens on. Valid values include ports 80, 443, and 1024 to 65535. The default value is port 80.
- **HTTPS Port:** The HTTPS port that the custom origin listens on. Valid values include ports 80, 443, and 1024 to 65535. The default value is port 443. If the value of **Origin Protocol Policy** is **HTTP Only**, you cannot update the value of **HTTPS Port**.

- d. Click **Create** if you're adding an origin, or click **Yes, Edit** if you're updating an existing origin.
5. To add or update a cache behavior for a download distribution, perform the following steps.
    - a. In the **Distribution Details** pane, click the **Behaviors** tab.
    - b. Click **Create Behavior** or **Edit**.
    - c. Enter or update the applicable values. For more information about the fields, see [Cache Behaviors \(p. 41\)](#).

- **Path Pattern:** The pattern (for example, *images/\* .jpg*) that specifies which requests you want this cache behavior to apply to.

#### **Note**

You cannot update the path pattern for the default cache behavior.

- **Origin:** The value of **Origin ID** for the origin that you want CloudFront to route requests to when a request matches the path pattern either for a cache behavior or for the default cache behavior.

- **Viewer Protocol Policy:** The protocol policy that you want viewers to use to access your content. If you specify **HTTP and HTTPS**, viewers can use both protocols. If you specify **HTTPS Only**, viewers are only allowed to access your content if they're using HTTPS.
- **Object Caching and Minimum TTL:** If your origin server is adding a `Cache-Control` header to your objects to control how long they stay in the CloudFront cache, select **Use Origin Cache Headers**.

To specify a minimum time that your objects stay in the CloudFront cache regardless of `Cache-Control` headers, select **Customize**. Then specify the minimum number of seconds that you want objects to stay in the CloudFront cache even if `Cache-Control` headers specify a lower value.

For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Object Expiration\)](#) (p. 65).

- **Forward Query Strings:** If your origin returns different versions of an object (for example, `images/image.jpg`) based on a query string in the URL, select **Yes**.

If your origin returns the same version of an object regardless of the query string, select **No**. This increases the likelihood that CloudFront can serve a request from the cache, which improves performance and reduces the load on your origin. For more information, see [How CloudFront Forwards and Logs Query String Parameters](#) (p. 62).

- d. Click **Create** if you're adding a cache behavior, or click **Yes, Edit** if you're updating an existing cache behavior.

6. To edit settings for a streaming distribution:

- a. In the **Distribution Details** pane, click **Edit**.
  - b. Change the applicable values.
- **Alternate Domain Names (CNAMEs):** Optional. Specify one or more domain names that you want to use for URLs for your objects instead of the domain name that CloudFront assigns when you create your distribution. For example, if you want the URL for the object:

`images/image.jpg`

to look like this:

`http://www.example.com/images/image.jpg`

instead of like this:

`http://d111111abcdef8.cloudfront.net/images/image.jpg`

you would create a CNAME for `www.example.com`. You can create up to 10 CNAMEs per distribution.

### Note

If you add a CNAME for `www.example.com` to your distribution, you also need to create (or update) a CNAME record with your DNS service to route queries for `www.example.com` to `d111111abcdef8.cloudfront.net`. For more information, see [Using Alternate Domain Names \(CNAMEs\)](#) (p. 53).

### Important

You must own any domain name that you specify as an alternate domain name.

- **Logging:** Optional. If you want CloudFront to log information about each request for an object and store the log files in an Amazon S3 bucket, select **On**, and specify the bucket and an optional prefix for the names of the log files. There is no extra charge to enable logging, but you accrue the usual Amazon S3 charges for storing and accessing the files. CloudFront doesn't delete the logs automatically, but you can delete them at any time. For more information, see [Access Logs \(p. 129\)](#).
- **Comments:** Optional. Enter any comments that you want to save with the distribution.
- **Distribution State:** Select **Enabled** if you want CloudFront to begin processing requests as soon as the distribution is created, or select **Disabled** if you want to wait until later.

c. Click **Yes, Edit**.

## Deleting a Distribution

You can create up to 100 download distributions and 100 streaming distributions for an AWS account. If you find that you no longer want to use a distribution, use the following procedure to delete it.

For information about deleting a download distribution using the CloudFront API, see [DELETE Distribution](#) in the *Amazon CloudFront API Reference*. For information about deleting a streaming distribution using the CloudFront API, see [DELETE Streaming Distribution](#) in the *Amazon CloudFront API Reference*.

### To Delete a CloudFront Distribution Using the CloudFront Console

1. Sign in to the AWS Management Console and open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. In the top pane of the CloudFront console, select the distribution that you want to delete.

If the value of the **State** column for the distribution is **Disabled** and the value of the **Status** column is:

- **Deployed**, skip to Step 8.
- **InProgress**, wait until the value changes to **Deployed**, then skip to Step 8.

If the value of the **State** column is **Enabled**, continue with Step 3.

3. Click the **General Configuration** tab.
4. Click **Edit General Configuration**.
5. Click **Disabled** to disable the distribution.
6. Click **Save** to save the change and close the dialog box.
7. Wait until the value of **Status** on the **General Configuration** tab changes to **Deployed**.
8. Click **Delete Distribution**.
9. Click **Yes, Delete**.
10. Click **Close**.

# Working with Objects

---

## Topics

- [Format of URLs for CloudFront Objects](#) (p. 60)
- [Adding, Removing, or Replacing Objects in a Distribution](#) (p. 64)
- [How CloudFront Processes Partial Requests for an Object \(Range GETs\)](#) (p. 71)
- [Specifying a Default Root Object \(Download Distributions Only\)](#) (p. 72)
- [Serving Compressed Files](#) (p. 74)
- [Restricting Access to Objects Based on the Geographic Location of End Users \(Geoblocking\)](#) (p. 76)

This section describes how you work with objects in the CloudFront system.

## Format of URLs for CloudFront Objects

### Topics

- [Public and Signed URLs](#) (p. 60)
- [How CloudFront Processes HTTP and HTTPS Requests](#) (p. 62)
- [How CloudFront Forwards and Logs Query String Parameters](#) (p. 62)

## Public and Signed URLs

When you create a distribution, you receive the CloudFront domain name associated with that distribution. You use that domain name when creating the links to your objects. If you have another domain name that you'd rather use (for example, `www.example.com`), you can add a CNAME alias. For more information, see [Using Alternate Domain Names \(CNAMEs\)](#) (p. 53).

When you create URLs to give end users access to objects in your CloudFront distribution, the URLs are either public URLs or signed URLs:

*Public URLs* allow users to access the following objects:

- Objects on which there are no restrictions.



- Objects in an Amazon S3 bucket that end users must access through CloudFront but that don't require a signed URL. These objects can't be accessed using an Amazon S3 URL.

*Signed URLs* are required to access the objects that are specified by a cache behavior that you have configured to require signed URLs. Note that if a request for an object (for example, `image.jpg`) matches the path patterns for two or more cache behaviors, CloudFront will process the request based on the cache behavior that is listed first in the distribution. If the first cache behavior doesn't require signed URLs and the second cache behavior does require signed URLs, end users will be able to access `image.jpg` without a signed URL.

For more information about cache behaviors, including path patterns, see [Cache Behaviors \(p. 41\)](#). For more information about signed URLs, see [Using a Signed URL to Serve Private Content \(p. 84\)](#).

## Public URLs for Objects in Amazon S3

A public URL for an object in an Amazon S3 bucket uses this format:

```
http://<CloudFront domain name>/<object name in Amazon S3 bucket>.
```

### Important

If the distribution serves streaming content, additional characters are required in the path to the file. For more information, see [Configuring the Media Player \(p. 51\)](#).

For example, suppose you have an Amazon S3 bucket called `mybucket`. The bucket contains a publicly readable object named `images/image.jpg`.

You create a CloudFront distribution and specify `mybucket.s3.amazonaws.com` as the origin server for this distribution. CloudFront returns `d111111abcdef8.cloudfront.net` as the domain name for the distribution and `EDFDVBD6EXAMPLE` as the distribution ID.

The URL you give to end users to access the object in this example is:

```
http://d111111abcdef8.cloudfront.net/images/image.jpg.
```

For more information about names and paths for Amazon S3 buckets, see [Virtual Hosting of Buckets](#) in the *Amazon Simple Storage Service Developer Guide*.

Anytime an end user accesses that object, CloudFront serves the object from the appropriate edge location. If the object isn't in that edge location, CloudFront goes to the origin server associated with the `EDFDVBD6EXAMPLE` distribution (`mybucket.s3.amazonaws.com`) and gets a copy of the object for the edge location to serve to the end user.

## Public URLs for Objects in a Custom Origin

The format of public URLs for objects in a custom origin are much like public URLs for objects in Amazon S3:

```
http://<CloudFront domain name>/<object name in custom origin>
```

If your object is in a folder on your origin server, then the CloudFront URL must include the name of the folder. For example, if `image.jpg` is located in the `images` folder, then the URL is:

```
http://d111111abcdef8.cloudfront.net/images/image.jpg
```

CloudFront gets objects from the domain that you specified when you created the distribution, using the object path and name in the public URL. For example, if the domain for your custom origin is `example.com`

and the object path and name is `/images/image.jpg`, CloudFront will get the object from the following location:

```
http://example.com/images/image.jpg
```

## How Public URLs Affect the Invalidation of Directories

If you use CloudFront URLs that give end users access to directories, we recommend that you always use the same URL format, either with a trailing slash (/) or without, for example:

```
http://d1111111abcdef8.cloudfront.net/images/
```

```
http://d1111111abcdef8.cloudfront.net/images
```

Browsers and other web applications will resolve both formats to the same directory. However, CloudFront stores public URLs exactly as you specify them, and if you want to invalidate a directory, you'll need to specify the exact same directory, including or excluding the slash. If you don't have a standard for how you specify directories, you'll need to invalidate the directory with and without the slash to ensure that CloudFront removes the directory from the edge location. If you've reached the limit for free invalidations for the month, you'll pay for both invalidations even though only one of the directories exists.

## Signed URLs

Signed URLs allow end users to access objects in a distribution that is configured to serve private content. The URLs include extra information that restricts access to the cached objects. For information about the format of signed URLs, see [Using a Signed URL to Serve Private Content \(p. 84\)](#).

## How CloudFront Processes HTTP and HTTPS Requests

For Amazon S3 origins, CloudFront accepts requests in both HTTP and HTTPS protocols for objects in a CloudFront distribution by default. CloudFront then passes the requests to your Amazon S3 bucket or custom origin using the same protocol with which the requests were made.

For custom origins, when you create your distribution, you can specify how CloudFront accesses your origin: HTTP only, or matching the protocol that is used by the viewer. For more information about how CloudFront handles HTTP and HTTPS requests for custom origins, see [Protocols \(p. 80\)](#).

For information about how to restrict your download distribution so that end users can only access objects using HTTPS, see [Using an HTTPS Connection to Access Your Objects \(p. 126\)](#). (This option doesn't apply to streaming distributions, which use the RTMP protocol.)

### Note

The charge for HTTPS requests is higher than the charge for HTTP requests. For more information about billing rates, go to the [CloudFront pricing plan](#).

## How CloudFront Forwards and Logs Query String Parameters

For download distributions, you can choose whether you want CloudFront to forward query string parameters to your origin. For streaming distributions, you cannot configure CloudFront to forward query string parameters to your origin.

For both types of distributions, if you enable logging, CloudFront logs the full URL, including query string parameters. For download distributions, this is true regardless of whether you have configured CloudFront to forward query strings. For more information about CloudFront logging, see [Access Logs](#) (p. 129).

For more information, see the applicable topic:

- [Query String Parameters and Download Distributions](#) (p. 63)
- [Query String Parameters and Streaming Distributions](#) (p. 64)

## Query String Parameters and Download Distributions

For download distributions, you can specify whether you want CloudFront to include query strings when it forwards requests to your origin. For example, you can specify whether you want CloudFront to forward the `?parameter1=a` part of the following URL:

`http://d1111111abcdef8.cloudfront.net/images/image.jpg?parameter1=a`

If you configure CloudFront to forward query strings to your origin, CloudFront will include the query string portion of the URL when caching the object. For example, the following query strings cause CloudFront to cache three objects. This is true even if your origin always returns the same `image.jpg` regardless of the query string:

- `http://d1111111abcdef8.cloudfront.net/images/image.jpg?parameter1=a`
- `http://d1111111abcdef8.cloudfront.net/images/image.jpg?parameter1=b`
- `http://d1111111abcdef8.cloudfront.net/images/image.jpg?parameter1=c`

If your origin returns different versions of an object (for example, `images/image.jpg`) based on the query string, select **Yes** for **Forward Query Strings** in the CloudFront console or specify `true` for the value of the `QueryString` element in the `DistributionConfig` complex type when you're using the CloudFront API.

If your origin returns the same version of an object regardless of the query string, select **No** or `false`. This increases the likelihood that CloudFront can serve a request from the cache, which improves performance and reduces the load on your origin.

The order of parameters matters in query strings. If you configure CloudFront to forward query strings to your origin, the following query strings cause CloudFront to cache two objects:

- `http://d1111111abcdef8.cloudfront.net/images/image.jpg?parameter1=a&parameter2=b`
- `http://d1111111abcdef8.cloudfront.net/images/image.jpg?parameter2=b&parameter1=a`

Case also matters in query strings. If you configure CloudFront to forward query strings to your origin, the following query strings cause CloudFront to cache two objects:

- `http://d1111111abcdef8.cloudfront.net/images/image.jpg?parameter1=a`
- `http://d1111111abcdef8.cloudfront.net/images/image.jpg?parameter1=A`

If you're using signed URLs to restrict access to your content (if you added trusted signers to your distribution), CloudFront removes the following query string parameters before forwarding the rest of the URL to CloudFront:

- Expires
- Key-Pair-Id
- Policy

- Signature

This means that if you're using signed URLs and you're configuring CloudFront to forward query string parameters to your origin, your own query string parameters cannot be named `Expires`, `Key-Pair-Id`, `Policy`, or `Signature`.

## Query String Parameters and Streaming Distributions

For streaming distributions, when CloudFront requests an object from the origin server, it removes any query string parameters. For example, if CloudFront receives the following request and `media.flv` is not already in the CloudFront cache:

```
http://d1111111abcdef8.cloudfront.net/media/media.flv?parameter1=a
```

it sends the following URL to your origin server:

```
http://d1111111abcdef8.cloudfront.net/media/media.flv
```

# Adding, Removing, or Replacing Objects in a Distribution

For information about adding objects to a distribution, see [Adding Objects to a CloudFront Distribution](#) (p. 64).

When you replace objects in your distribution, we recommend that you use versioned object names. For more information, see [Updating Existing Objects Using Versioned Object Names](#) (p. 64). You can also replace objects with objects that have the same name. See [Updating Existing Objects Using the Same Object Names](#) (p. 65). Regardless of how you choose to replace objects in your distribution, we recommend that you specify when objects should be removed from the CloudFront cache. For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Object Expiration\)](#) (p. 65).

If you need to quickly remove objects from a distribution, you can invalidate them. For more information, see [Invalidating Objects](#) (p. 68).

## Adding Objects to a CloudFront Distribution

When you add an object to your origin, ensure that you're adding it to one of the Amazon S3 buckets in your distribution or, for a custom origin, to a directory in the specified domain.

When you add an object to your origin and expose a CloudFront link to the object, a CloudFront edge location won't fetch the object from the origin until the edge location receives an end-user request for the object.

CloudFront servers don't determine the MIME type for the objects they serve. When you upload an object to your origin, you should set the `Content-Type` header field for the object.

## Updating Existing Objects Using Versioned Object Names

When you update existing objects in a CloudFront distribution, we recommend that you include some sort of version identifier either in your object names or in your directory names to give yourself better

control over your content. This identifier might be a date-time stamp, a sequential number, or some other method of distinguishing two versions of the same object.

For example, instead of naming a graphic file `image.jpg`, you might call it `image_1.jpg`. When you want to start serving a new version of the file, you'd name the new file `image_2.jpg`, and you'd update your links to point to `image_2.jpg`. Alternatively, you might put all graphics in an `images_v1` directory and, when you want to start serving new versions of one or more graphics, you'd create a new `images_v2` directory, and you'd update your links to point to that directory. With versioning, you don't have to wait for an object to expire before CloudFront begins to serve a new version of it, and you don't have to pay for object invalidation.

Even if you version your objects, we still recommend that you set an expiration date. For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Object Expiration\)](#) (p. 65).

#### **Note**

Specifying versioned object names or directory names is not related to Amazon S3 object versioning.

## **Updating Existing Objects Using the Same Object Names**

Although you can update existing objects in a CloudFront distribution and use the same object names, we don't recommend it. CloudFront distributes objects to edge locations only when the objects are requested, not when you put new or updated objects in your origin. If you update an existing object in your origin with a newer version that has the same name, an edge location won't get that new version from your origin until both of the following occur:

- The old version of the object in the cache expires. For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Object Expiration\)](#) (p. 65).
- There's an end user request for the object at that edge location.

If you use the same names when you replace objects, you can't control when CloudFront starts to serve the new files. By default, CloudFront caches objects in edge locations for 24 hours. (For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Object Expiration\)](#) (p. 65).) For example, if you're replacing all of the objects on an entire website:

- Objects for the less popular pages may not be in any edge locations. The new versions of these objects will start being served on the next request.
- Objects for some pages may be in some edge locations and not in others, so your end users will see different versions depending on which edge location they're served from.
- New versions of the objects for the most popular pages might not be served for up to 24 hours because CloudFront might have retrieved the objects for those pages just before you replaced the objects with new versions.

## **Specifying How Long Objects Stay in a CloudFront Edge Cache (Object Expiration)**

You can control how long your objects stay in a CloudFront cache before CloudFront forwards another request to your origin. Reducing the duration allows you to serve dynamic content. Increasing the duration means your customers get better performance because your objects are more likely to be served directly from the edge cache. A longer duration also reduces the load on your origin.

Typically, CloudFront serves an object from an edge location until the object expires. After it expires, the next time the edge location gets an end-user request for the object, CloudFront forwards the request to the origin server to verify that the cache contains the latest version of the object:

- If CloudFront already has the latest version, the origin returns only a 304 status code (not modified).
- If CloudFront does not have the latest version, the origin returns a 200 status code (OK) and the latest version of the object.

If an object in an edge location isn't frequently requested, CloudFront might evict the object—remove the object before its expiration date—to make room for objects that are more popular.

By default, each object automatically expires after 24 hours. To specify a different expiration time, configure your origin to add a value for either the `Cache-Control max-age` directive or the `Expires` header field to each object:

- The `Cache-Control max-age` directive lets you specify how long (in seconds) you want the object to remain in the cache before CloudFront gets the object again from the origin server. The minimum expiration time CloudFront supports is 0 seconds and the maximum is in the year 2038.
- The `Expires` header field lets you specify an expiration date and time using the format specified in [RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1 Section 3.3.1, Full Date](#), for example:

```
Sat, 30 Jun 2012 23:59:59 GMT
```

### Important

After the expiration date and time passes, CloudFront gets the object again from the origin server every time an edge location receives a request for the object.

We recommend that you use the `Cache-Control max-age` directive instead of the `Expires` header field to control object caching. If you specify values both for `Cache-Control max-age` and for `Expires`, CloudFront uses only the value of `max-age`.

You cannot use the HTTP `Cache-Control` or `Pragma` header fields in a `GET` request from an end user to force CloudFront to go back to the origin server for the object. CloudFront ignores those header fields from the end user.

For more information about the `Cache-Control` and `Expires` header fields, see the following sections in *RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1*:

- [Section 14.9 Cache Control](#)
- [Section 14.21 Expires](#)

For an example of how to add `Cache-Control` and `Expires` header fields using the AWS SDK for PHP, see [Upload an Object Using the AWS SDK for PHP](#) in the *Amazon Simple Storage Service Developer Guide*. Some third-party tools are also able to add these fields.

## Specifying the Minimum Time that CloudFront Caches Objects for Download Distributions

For download distributions, if you are adding `Cache-Control` or `Expires` headers to your objects, you can also specify the minimum amount of time that CloudFront keeps an object in the cache before forwarding another request to the origin. Here's how object headers and a minimum TTL control how long objects remain in the CloudFront cache:

**Amazon CloudFront Developer Guide**  
**Specifying How Long Objects Stay in a CloudFront Edge**  
**Cache (Object Expiration)**

	Minimum TTL = 0 (Default)	Minimum TTL > 0
<b>Origin adds <code>Cache-Control max-age</code> directive to objects</b>	Objects are cached for the value of the <code>Cache-Control max-age</code> directive	Objects are cached for the greater of the value of the <code>Cache-Control max-age</code> directive or the value of the CloudFront Minimum TTL
<b>Origin does not add <code>Cache-Control max-age</code> directive</b>	Objects are cached for 24 hours.	Objects are cached for 24 hours.
<b>Origin adds <code>Expires</code> header</b>	Objects are cached until the date in the <code>Expires</code> header. After the date passes, CloudFront forwards every request to the origin.	Objects are cached until the date in the <code>Expires</code> header. After the date passes, CloudFront forwards every request to the origin.
<b>Origin adds <code>Cache-Control no-cache, no-store, and/or private</code> directives to objects</b>	CloudFront respects the headers	Objects are cached for the Minimum TTL

For information about how to change settings for download distributions using the CloudFront console, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 55\)](#). For information about how to change settings for download distributions using the CloudFront API, see [PUT Config](#).

## Adding Headers to Your Objects Using the Amazon S3 Console

To add a `Cache-Control` or `Expires` header field to Amazon S3 objects using the Amazon S3 console

1. Sign in to the AWS Management Console and open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
2. In the Amazon S3 console, in the Buckets pane, click the name of the bucket that contains the files.
3. In the **Objects and Folders** pane, select the first object to which you want to add a header field.
4. At the top of the **Objects and Folders** pane, click **Actions** and click **Properties**.
5. In the **Properties** pane, click the **Metadata** tab.
6. On the Metadata tab, click **Add More Metadata**.
7. In the **Key** list, click **Cache-Control** or **Expires**, as applicable.
8. In the **Value** field, enter the applicable value:
  - For a `Cache-Control` field, enter:  
`max-age=`*number of seconds that you want objects to stay in a CloudFront edge cache*
  - For an **Expires** field, enter a date and time in HTML format.
9. Click **Save**.
10. If you want to add a header field to additional objects, in the **Objects and Folders** pane, click the name of the next object, and repeat Steps 6 through 9.



## Invalidating Objects

### Topics

- [Choosing Between Invalidating Objects and Using Versioned Object Names \(p. 68\)](#)
- [Invalidating Objects and Displaying Information about Invalidations \(p. 68\)](#)
- [Third-Party Tools for Invalidating Objects \(p. 71\)](#)
- [Invalidation Limits \(p. 71\)](#)
- [Paying for Object Invalidation \(p. 71\)](#)

If you need to remove an object from CloudFront edge-server caches before it would expire, you can do one of the following:

- Invalidate the object. The next time an end user requests the object, CloudFront returns to the origin to fetch the latest version of the object.
- Use object versioning to serve a different version of the object that has a different name. For more information, see [Updating Existing Objects Using Versioned Object Names \(p. 64\)](#).

You can invalidate a specified number of objects each month for free. Above that limit, you pay a fee for each object that you invalidate. For example, to invalidate a directory and all of the files in the directory, you must invalidate the directory and each file individually. If you need to invalidate a lot of files, it may be easier and less expensive to create a new distribution and change your object paths to refer to the new distribution. For more information about the charges for invalidation, see [Paying for Object Invalidation \(p. 71\)](#).

## Choosing Between Invalidating Objects and Using Versioned Object Names

To control the versions of objects served from your distribution, you can either invalidate objects or give them versioned file names. If you'll want to update your objects frequently, we recommend that you primarily use object versioning for the following reasons:

- Versioning enables you to control which object a request returns even when the end user has a version cached either locally or behind a corporate caching proxy. If you invalidate the object, the end user may continue to see the old version until it expires from those caches.
- File names are included in the CloudFront access logs, so versioning makes it easier to analyze the results of object changes.
- Versioning provides a way to serve different versions of objects to different end users.
- Versioning simplifies rolling forward and back between object revisions.
- Versioning is less expensive. You still have to pay for CloudFront to transfer new versions of your objects to edge locations, but you don't have to pay the per-file charge for invalidating objects.

For more information about object versioning, see [Updating Existing Objects Using Versioned Object Names \(p. 64\)](#).

## Invalidating Objects and Displaying Information about Invalidations

You can use the CloudFront console or CloudFront API actions to create and run an invalidation, display a list of the invalidations that you submitted previously, and display detailed information about an individual invalidation. You can also copy an existing invalidation, edit the list of object paths, and run the edited invalidation.



See the applicable topic:

- [Invalidating Objects Using the CloudFront Console \(p. 69\)](#)
- [Copying, Editing, and Rerunning an Existing Invalidation Using the CloudFront Console \(p. 69\)](#)
- [Listing Invalidations Using the CloudFront Console \(p. 70\)](#)
- [Displaying Information About an Invalidation Using the CloudFront Console \(p. 70\)](#)
- [Invalidating Objects and Displaying Information about Invalidations Using the CloudFront API \(p. 71\)](#)

## Invalidating Objects Using the CloudFront Console

You can create any number of invalidations, but you can have only three invalidation per distribution in progress at one time. Each request can contain up to 1000 objects to invalidate. If you exceed these limits, the CloudFront console displays an error message. To determine how many invalidations are currently in progress, see the **Status** column on the **Invalidations** tab.

### To invalidate objects using the CloudFront console

1. Sign in to the AWS Management Console and open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Click the distribution for which you want to invalidate objects.
3. Click **Distribution Settings**.
4. Click the **Invalidations** tab.
5. Click **Create Invalidation**.
6. Enter the paths of the objects that you want to invalidate. Note the following:
  - The path is relative to the distribution. A leading / is optional. For example, to invalidate the object at `http://d1111111abcdef8.cloudfront.net/images/image2.jpg`, you would specify:  
  
`/images/image2.jpg`  
  
or  
  
`images/image2.jpg`
  - To invalidate the default root object, specify the path to the object the same way you specify the path to any other object.
  - You can specify up to 1000 objects.
  - If the object is a directory and if you have not standardized on a method for specifying directories—with or without a trailing slash (/)—we recommend that you invalidate the directory both with and without a trailing slash, for example, `images` and `images/`. For more information, see [How Public URLs Affect the Invalidation of Directories \(p. 62\)](#).
  - If the path includes non-ASCII characters or unsafe characters as defined in RFC 1783 (<http://www.ietf.org/rfc/rfc1738.txt>), URL-encode those characters. Do not URL-encode any other characters in the path, or CloudFront will not invalidate the old version of the updated object.
7. Click **Invalidate**.

## Copying, Editing, and Rerunning an Existing Invalidation Using the CloudFront Console

You can copy an invalidation that you created previously, update the list of object paths, and run the updated validation. You cannot copy an existing invalidation, update the object paths, and save the updated invalidation without running it.

## Important

If you copy an invalidation that is still in progress, update the list of object paths, and run the updated invalidation, CloudFront will not stop or delete the invalidation that you copied. If any object paths appear in the original and in the copy, CloudFront will try to invalidate the objects twice, and both invalidations will count against your maximum number of free invalidations for the month. If you've already reached the maximum number of free validations, you'll be charged for both invalidations of each object. For more information, see [Invalidation Limits \(p. 71\)](#)

### To copy, edit, and rerun an existing invalidation using the CloudFront console

1. Sign in to the AWS Management Console and open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Click the distribution that contains the invalidation that you want to copy.
3. Click **Distribution Settings**.
4. Click the **Invalidations** tab.
5. Click the invalidation that you want to copy.

If you aren't sure which invalidation you want to copy, you can click an invalidation and click **Details** to display detailed information about that invalidation.

6. Click **Copy**.
7. Update the list of object paths if applicable.
8. Click **Invalidate**.

## Listing Invalidations Using the CloudFront Console

Using the console, you can display a list of the last 100 invalidations that you've created and run for a distribution. If you want to get a list of more than 100 invalidations, use the GET Invalidation List API action. For more information, go to [GET Invalidation List](#) in the *Amazon CloudFront API Reference*.

### To list invalidations using the CloudFront console

1. Sign in to the AWS Management Console and open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Click the distribution for which you want to display a list of invalidations.
3. Click **Distribution Settings**.
4. Click the **Invalidations** tab.

## Displaying Information About an Invalidation Using the CloudFront Console

You can display detailed information about an invalidation, including distribution ID, invalidation ID, the status of the invalidation, the date and time that the invalidation was created, and a complete list of the object paths.

### To display information about an invalidation using the CloudFront console

1. Sign in to the AWS Management Console and open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
2. Click the distribution that contains the invalidation about which you want to display detailed information.
3. Click **Distribution Settings**.
4. Click the **Invalidations** tab.
5. Click the invalidation about which you want to display detailed information.
6. Click **Details**.

## Invalidating Objects and Displaying Information about Invalidations Using the CloudFront API

For information about invalidating objects and about displaying information about invalidations using the CloudFront API, see the applicable topic in the *Amazon CloudFront API Reference*:

- Invalidating objects: [POST Invalidation](#)
- Getting a list of your invalidations: [GET Invalidation List](#)
- Getting information about a specific invalidation: [GET Invalidation](#)

## Third-Party Tools for Invalidating Objects

In addition to the invalidation methods provided by CloudFront, several third-party tools provide ways to invalidate objects. For a list of tools, see [Invalidating Objects](#) (p. 219).

## Invalidation Limits

You can make any number of invalidation requests, but you can have only three invalidation requests per distribution in progress at one time. Each request can contain up to 1000 objects to invalidate. If you exceed these limits, CloudFront returns an error message.

### Note

It usually takes 10 to 15 minutes for CloudFront to complete your invalidation request, depending on the size of the request.

## Paying for Object Invalidation

The first 1000 object invalidations you request per month are free; you pay for each object invalidation over 1000 in a month. This limit applies to the total number of object invalidations across all of the distributions that you create with one AWS account. For example, if you use the AWS account `john@example.com` to create three distributions, and each distribution has 600 object invalidations in a given month (for a total of 1,800 invalidations), AWS will charge you for 800 object invalidations in that month. For specific information about invalidation pricing, go to [Amazon CloudFront Pricing](#).

### Note

For the purposes of invalidation pricing, an object invalidation request is defined as a single `Path` element object. For more information about the `Path` element, see [Invalidating Objects and Displaying Information about Invalidations](#) (p. 68).

# How CloudFront Processes Partial Requests for an Object (Range GETs)

For a large object, an end user's browser or client might make multiple `GET` requests and use the `Range` request header to download the object in smaller units. These requests for ranges of bytes, sometimes known as `Range GET` requests, improve the efficiency of partial downloads and the recovery from partially failed transfers.

When CloudFront receives a `Range GET` request, it checks the cache in the edge location that received the request. If the cache in that edge location already contains the entire object or the requested portion

of the object, CloudFront immediately serves the requested range from the cache. If the cache doesn't contain the requested range, CloudFront forwards the request to the origin.

If the origin supports `Range` GET requests, it returns the requested range; otherwise, it returns the entire object. If the origin returns a range, CloudFront serves the requested range and also caches it for future requests. If the origin returns the entire object, CloudFront serves the entire object and also caches the entire object for future requests. (Amazon S3 supports `Range` GET requests, as do some HTTP servers, for example, Apache and IIS. For information about whether your HTTP server does, go to the documentation for your HTTP server.)

CloudFront generally follows the RFC specification for the `Range` header. However, if your `Range` headers don't adhere to the following requirements, CloudFront will return HTTP status code 200 with the full object instead of status code 206 with the specified ranges:

- The ranges must be listed in ascending order. For example, `100-200, 300-400` is valid, `300-400, 100-200` is not valid.
- The ranges must not overlap. For example, `100-200, 150-250` is not valid.
- All of the ranges specifications must be valid. For example, you can't specify a negative value as part of a range.

For more information about the `Range` request header, see "Section 14.35 Range" in *Hypertext Transfer Protocol -- HTTP/1.1* at <http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.35>.

## Specifying a Default Root Object (Download Distributions Only)

You can configure CloudFront to return a specific object (the default root object) when an end user requests the root URL for your distribution instead of an object in your distribution. Specifying a default root object avoids exposing the contents of your distribution.

For example, the following request points to the object `image.jpg`:

```
http://d1111111abcdef8.cloudfront.net/image.jpg
```

The following request points to the root URL of the same distribution instead of to a specific object:

```
http://d1111111abcdef8.cloudfront.net/
```

When you define a default root object, an end-user request that calls the root of your distribution returns the default root object. For example, if you designate the file `index.html` as your default root object, a request for:

```
http://d1111111abcdef8.cloudfront.net/
```

returns:

```
http://d1111111abcdef8.cloudfront.net/index.html
```

However, if you define a default root object, an end-user request for a subdirectory of your distribution does not return the default root object. For example, suppose `index.html` is your default root object and that CloudFront receives an end-user request for the `install` directory under your CloudFront distribution:

```
http://d1111111abcdef8.cloudfront.net/install/
```

CloudFront will not return the default root object even if a copy of `index.html` appears in the `install` directory.

The behavior of CloudFront default root objects is different from the behavior of Amazon S3 index documents. When you configure an Amazon S3 bucket as a website and specify the index document, Amazon S3 returns the index document even if a user requests a subdirectory in the bucket. (A copy of the index document must appear in every subdirectory.) For more information about configuring Amazon S3 buckets as websites and about index documents, see the [Hosting Websites on Amazon S3](#) chapter in the *Amazon Simple Storage Service Developer Guide*.

### Important

Remember that a default root object applies only to your CloudFront distribution. You still need to manage security for your origin. For example, if you are using an Amazon S3 origin, you still need to set your Amazon S3 bucket ACLs appropriately to ensure the level of access you want on your bucket.

If you don't define a default root object, requests for the root of your distribution pass to your origin server. If you are using an Amazon S3 origin, any of the following might be returned:

- **A list of the contents of your Amazon S3 bucket**—Under any of the following conditions, the contents of your origin are visible to anyone who uses CloudFront to access your distribution:
  - Your bucket is not properly configured.
  - The Amazon S3 permissions on the bucket associated with your distribution and on the objects in the bucket grant access to *everyone*.
  - An end user accesses your origin using your origin root URL.
- **A list of the private contents of your origin**—If you configure your origin as a private distribution (only you and CloudFront have access), the contents of the Amazon S3 bucket associated with your distribution are visible to anyone who has the credentials to access your distribution through CloudFront. In this case, users are not able to access your content through your origin root URL. For more information about distributing private content, see [Using a Signed URL to Serve Private Content \(p. 84\)](#).
- **Error 403 Forbidden**—CloudFront returns this error if the permissions on the Amazon S3 bucket associated with your distribution or the permissions on the objects in that bucket deny access to CloudFront and to everyone.

To avoid exposing the contents of your download distribution or returning an error, perform the following procedure to specify a default root object for your distribution.

### To specify a default root object for your distribution

1. Upload the default root object to the origin that your distribution points to.

The file can be any type supported by CloudFront. For a list of constraints on the file name, see the description of the `DefaultRootObject` element in [DistributionConfig Complex Type](#).

#### Note

If the file name of the default root object is too long or contains an invalid character, CloudFront returns the error `HTTP 400 Bad Request - InvalidDefaultRootObject`. In addition, CloudFront caches the code for five minutes and writes the results to the access logs.

2. Confirm that the permissions for the object grant CloudFront at least `read` access.

For more information about Amazon S3 permissions, see [Access Control](#) in the *Amazon Simple Storage Service Developer Guide*. For information on using the Amazon S3 console to update permissions, go to the [Amazon Simple Storage Service Console User Guide](#).

3. Update your distribution to refer to the default root object using the CloudFront console or the CloudFront API.

To specify a default root object using the CloudFront console:

- a. Sign in to the AWS Management Console and open the Amazon CloudFront console at <https://console.aws.amazon.com/cloudfront/>.
- b. In the list of distributions in the top pane, select the distribution to update.
- c. In the **Distribution Details** pane, on the **General** tab, click **Edit**.
- d. In the **Edit Distribution** dialog box, in the **Default Root Object** field, enter the file name of the default root object.
- e. To save your changes, click **Yes, Edit**.

To update your configuration using the CloudFront API, you specify a value for the `DefaultRootObject` element in your distribution. For information about using the CloudFront API to specify a default root object, see [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.

4. Confirm that you have enabled the default root object by requesting your root URL. If your browser doesn't display the default root object, perform the following steps:
  - a. Confirm that your distribution is fully deployed by viewing the status of your distribution in the CloudFront console.
  - b. Repeat Steps 2 and 3 to verify that you granted the correct permissions and that you correctly updated the configuration of your distribution to specify the default root object.

## Serving Compressed Files

Amazon CloudFront can serve both compressed and uncompressed files from an origin server. CloudFront relies on the origin server either to compress the files or to have compressed and uncompressed versions of files available; CloudFront does not perform the compression on behalf of the origin server. With some qualifications, CloudFront can also serve compressed content from Amazon S3. For more information, see [Choosing the File Types to Compress](#) (p. 76).

Serving compressed content makes downloads faster because the files are smaller—in some cases, less than half the size of the original. Especially for JavaScript and CSS files, faster downloads translates into faster rendering of web pages for your users. In addition, because the cost of CloudFront data transfer is based on the total amount of data served, serving compressed files is less expensive than serving uncompressed files.

CloudFront can only serve compressed data if the viewer (for example, a web browser or media player) requests compressed content by including `Accept-Encoding: gzip, deflate` in the request header. The content must be compressed using gzip or deflate; other compression algorithms are not supported. If the request header includes additional content encodings, for example, `sdch`, CloudFront removes them before forwarding the request to the origin server. If either `gzip` or `deflate` is missing from the `Accept-Encoding` field, CloudFront serves only the uncompressed version of the file. For more information about the `Accept-Encoding` request-header field, see "Section 14.3 Accept Encoding" in *Hypertext Transfer Protocol -- HTTP/1.1* at <http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html>.

## How CloudFront Serves Compressed Content from a Custom Origin

Here's how CloudFront commonly serves compressed content from a custom origin to a web application:

1. You configure your web server to compress selected file types. For more information, see [Choosing the File Types to Compress \(p. 76\)](#).
2. You create a CloudFront distribution.
3. You program your web application to access files using CloudFront URLs.
4. A user accesses your application in a web browser.
5. CloudFront directs web requests to the edge location that has the lowest latency for the user, which may or may not be the geographically closest edge location.
6. At the edge location, CloudFront checks the cache for the object referenced in each request. If the browser included `Accept-Encoding: gzip, deflate` in the request header, CloudFront checks for a compressed version of the file. If not, CloudFront checks for an uncompressed version.
7. If the file is in the cache, CloudFront returns the file to the web browser. If the file is not in the cache:
  - a. CloudFront forwards the request to the origin server.
  - b. If the request is for a type of file that you want to serve compressed (see Step 1), the web server compresses the file.
  - c. The web server returns the file (compressed or uncompressed, as applicable) to CloudFront.
  - d. CloudFront adds the file to the cache and serves the file to the user's browser.

## Serving Compressed Files When Your Origin Server Is Running IIS

By default, IIS does not serve compressed content for requests that come through proxy servers such as CloudFront. If you're using IIS and if you configured IIS to compress content by using the `httpCompression` element, change the values of the `noCompressionForHttp10` and `noCompressionForProxies` attributes to `false`.

In addition, if you have compressed objects that are requested less frequently than every few seconds, you may have to change the values of `frequentHitThreshold` and `frequentHitTimePeriod`.

For more information, refer to the IIS documentation on the Microsoft website.

## Serving Compressed Files from Amazon S3

If you want to serve compressed files from Amazon S3:

1. Create two versions of each file, one compressed and one uncompressed. To ensure that the compressed and uncompressed versions of a file don't overwrite one another in the CloudFront cache, give each file a unique name, for example, `welcome.js` and `welcome.js.gz`.
2. Open the Amazon S3 console at <https://console.aws.amazon.com/s3/>.
3. Upload both versions to Amazon S3.
4. Add a `Content-Encoding` header field for each compressed file and set the field value to `gzip` or `deflate`, as applicable.

For an example of how to add a `Content-Encoding` header field using the AWS SDK for PHP, see [Upload an Object Using the AWS SDK for PHP](#) in the *Amazon Simple Storage Service Developer Guide*. Some third-party tools are also able to add this field.

To add a `Content-Encoding` header field and set the field value using the Amazon S3 console, perform the following procedure:



- a. In the Amazon S3 console, in the Buckets pane, click the name of the bucket that contains the compressed files.
  - b. At the top of the Objects and Folders pane, click **Actions** and, in the **Actions** list, click **Properties**.
  - c. In the Properties pane, click the **Metadata** tab.
  - d. In the Objects and Folders pane, click the name of a file for which you want to add a `Content-Encoding` header field.
  - e. On the Metadata tab, click `Add More Metadata`.
  - f. In the Key list, click `Content-Encoding`.
  - g. In the Value field, enter `gzip` or `deflate`, as applicable.
  - h. Click **Save**.
  - i. Repeat Step 4d through 4h for the remaining compressed files.
5. When generating HTML that links to content in CloudFront (for example, using php, asp, or jsp), evaluate whether the request from the viewer includes `Accept-Encoding: gzip, deflate` in the request header. If so, rewrite the corresponding link to point to the compressed object name.

## Choosing the File Types to Compress

Some types of files compress well, for example, HTML, CSS, and JavaScript files. Some types of files may compress a few percent, but not enough to justify the additional processor cycles required for your web server to compress the content, and some types of files even get larger when they're compressed. File types that generally don't compress well include graphic files that are already compressed (.jpg, .gif), video formats, and audio formats. We recommend that you test compression for the file types in your distribution to ensure that there is sufficient benefit to compression.

## Restricting Access to Objects Based on the Geographic Location of End Users (Geoblocking)

You can restrict access to your objects based on the geographic location of your end users by using CloudFront's private-content feature along with a third-party geolocation service. To restrict access, you make your CloudFront distribution private. When the end user requests an object in your private distribution, you send the user's IP address to the geolocation service of your choice. Based on the geographic location returned by the third-party geolocation service for that end user, your web application either generates a CloudFront signed URL for the user or displays a message explaining why the user isn't allowed to access the object.

For more information about geoblocking, see the tutorial [Restricting Access to Files in a CloudFront Distribution Based on Geographic Location \(Geoblocking\)](#) (p. 191). For more information about CloudFront private content, see [Using a Signed URL to Serve Private Content](#) (p. 84).



# Request and Response Behavior, and Supported HTTP Status Codes

---

See the applicable section:

- [Request and Response Behavior for Amazon S3 Origins \(p. 77\)](#)
- [Request and Response Behavior, and Supported HTTP Status Codes for Custom Origins \(p. 79\)](#)

## Request and Response Behavior for Amazon S3 Origins

### Topics

- [How CloudFront Processes and Forwards Requests to Your Amazon S3 Origin Server \(p. 77\)](#)
- [How CloudFront Processes Responses from Your Amazon S3 Origin Server \(p. 79\)](#)

## How CloudFront Processes and Forwards Requests to Your Amazon S3 Origin Server

For information about how CloudFront processes end-user requests and forwards the requests to your Amazon S3 origin, see the applicable topic:

- [HTTP Methods \(p. 78\)](#)
- [Query Strings \(p. 78\)](#)
- [Protocols \(p. 78\)](#)
- [IP Addresses \(p. 78\)](#)
- [Caching Duration and Minimum TTL \(p. 78\)](#)
- [Conditional GETs \(p. 78\)](#)

## HTTP Methods

CloudFront accepts only `GET` and `HEAD` requests from end users.

## Query Strings

For download distributions, you can configure whether CloudFront forwards query string parameters to your Amazon S3 origin. For streaming distributions, CloudFront does not forward query string parameters. For more information, see [How CloudFront Forwards and Logs Query String Parameters \(p. 62\)](#).

## Protocols

CloudFront forwards HTTP or HTTPS requests to the origin server based on the protocol of the request that the end user sends to CloudFront, either HTTP or HTTPS.

## IP Addresses

The IP address that CloudFront forwards to Amazon S3 is the IP address of a CloudFront server, not the IP address of the end user's computer.

## Caching Duration and Minimum TTL

For download distributions, to control how long your objects stay in a CloudFront cache before CloudFront forwards another request to your origin, you can:

- Configure your origin to add a `Cache-Control` or an `Expires` header field to each object.
- Specify a value for Minimum TTL in CloudFront cache behaviors.
- Use the default value of 24 hours.

For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Object Expiration\) \(p. 65\)](#).

## Conditional GETs

When CloudFront receives a request for an object that has expired from an edge cache, it forwards the request to the Amazon S3 origin either to get the latest version of the object or to get confirmation from Amazon S3 that the CloudFront edge cache already has the latest version. When Amazon S3 originally sent the object to CloudFront, it included an `ETag` value and a `LastModified` value in the response. In the new request that CloudFront forwards to Amazon S3, CloudFront adds one of the following:

- An `If-Match` or `If-None-Match` header that contains the `ETag` value for the expired version of the object.
- An `If-Modified-Since` header that contains the `LastModified` value for the expired version of the object.

Amazon S3 uses this information to determine whether the object has been updated and, therefore, whether to return the entire object to CloudFront or to return only an HTTP 304 status code (not modified).

## How CloudFront Processes Responses from Your Amazon S3 Origin Server

The maximum size of a response body that CloudFront will return to the end user is 20 GB. This includes chunked transfer responses that don't specify the `Content-Length` header value.

## Request and Response Behavior, and Supported HTTP Status Codes for Custom Origins

### Topics

- [How CloudFront Processes and Forwards Requests to Your Custom Origin Server \(p. 79\)](#)
- [How CloudFront Processes Responses from Your Custom Origin Server \(p. 81\)](#)
- [Supported HTTP Status Codes for Custom Origin Servers \(p. 83\)](#)

## How CloudFront Processes and Forwards Requests to Your Custom Origin Server

For information about how CloudFront processes end-user requests and forwards the requests to your custom origin, see the applicable topic:

- [HTTP Methods \(p. 79\)](#)
- [Query Strings \(p. 79\)](#)
- [HTTP Version \(p. 80\)](#)
- [Encryption \(p. 80\)](#)
- [Protocols \(p. 80\)](#)
- [Client Authentication \(p. 80\)](#)
- [Removed Header Fields \(p. 80\)](#)
- [IP Addresses \(p. 80\)](#)
- [Compression \(p. 80\)](#)
- [Cookies \(p. 81\)](#)
- [Caching Duration and Minimum TTL \(p. 81\)](#)
- [Conditional GETs \(p. 81\)](#)

## HTTP Methods

CloudFront accepts only `GET` and `HEAD` requests from end users.

## Query Strings

You can configure whether CloudFront forwards query string parameters to your origin. For more information, see [How CloudFront Forwards and Logs Query String Parameters \(p. 62\)](#).

## HTTP Version

CloudFront forwards requests to your custom origin using HTTP/1.0, but supports most of the HTTP 1.1 specification. To enhance performance, we recommend that you include the `Keep-Alive` header in end-user requests.

## Encryption

CloudFront forwards HTTPS requests to the origin server using the RC4-MD5 cipher.

## Protocols

CloudFront forwards HTTP or HTTPS requests to the origin server based on the following:

- The protocol of the request that the end user sends to CloudFront, either HTTP or HTTPS.
- The value of the **Origin Protocol Policy** field in the CloudFront console or, if you're using the CloudFront API, the `OriginProtocolPolicy` element in the `DistributionConfig` complex type. In the CloudFront console, the options are **HTTP Only** and **Match Viewer**.

If you specify **HTTP Only**, CloudFront forwards requests to the origin server using only the HTTP protocol, regardless of the protocol in the end-user request.

If you specify **Match Viewer**, CloudFront forwards requests to the origin server using the protocol in the end-user request.

### Caution

If the end-user request uses the HTTPS protocol, and if the origin server returns an invalid certificate or a self-signed certificate, CloudFront drops the TCP connection.

If you aren't sure which protocol to use, we recommend that you specify HTTP only.

For information about how to update a distribution using the CloudFront console, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 55\)](#). For information about how to update a distribution using the CloudFront API, see [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.

## Client Authentication

Do not configure your origin server to request client authentication. When CloudFront forwards an end-user request to your origin server over HTTPS, CloudFront does not present a certificate.

## Removed Header Fields

CloudFront removes hop-by-hop header fields such as the `Authorization` and `Connection` fields before forwarding requests to your origin.

## IP Addresses

The IP address that CloudFront forwards to the origin server is the IP addresses of a CloudFront server, not the IP address of the end user's computer.

## Compression

CloudFront forwards requests that have the `Accept-Encoding` field values `"identity"` and `"gzip, deflate"`. CloudFront accepts `"deflate, gzip"` and reorders the values to `"gzip, deflate"`.

For more information, see [Serving Compressed Files \(p. 74\)](#).

## Cookies

CloudFront does not forward `Cookie` header fields to your origin. In addition, if the origin server includes `Set-Cookie` header fields, CloudFront removes these headers before caching the objects at the edge location and before serving the objects to the end user.

## Caching Duration and Minimum TTL

For download distributions, to control how long your objects stay in a CloudFront cache before CloudFront forwards another request to your origin, you can:

- Configure your origin to add a `Cache-Control` or an `Expires` header field to each object.
- Specify a value for Minimum TTL in CloudFront cache behaviors.
- Use the default value of 24 hours.

For more information, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Object Expiration\) \(p. 65\)](#).

## Conditional GETs

When CloudFront receives a request for an object that has expired from an edge cache, it forwards the request to the origin either to get the latest version of the object or to get confirmation from the origin that the CloudFront edge cache already has the latest version. Typically, when the origin last sent the object to CloudFront, it included an `ETag` value, a `LastModified` value, or both values in the response. In the new request that CloudFront forwards to the origin, CloudFront adds one of the following:

- An `If-Match` or `If-None-Match` header that contains the `ETag` value for the expired version of the object.
- An `If-Modified-Since` header that contains the `LastModified` value for the expired version of the object.

The origin uses this information to determine whether the object has been updated and, therefore, whether to return the entire object to CloudFront or to return only an HTTP 304 status code (not modified).

## How CloudFront Processes Responses from Your Custom Origin Server

For information about how CloudFront processes responses from custom origin servers, see the applicable topic:

- [Maximum File Size \(p. 82\)](#)
- [Caching \(p. 82\)](#)
- [Content Negotiation \(p. 82\)](#)
- [Redirects \(p. 82\)](#)
- [Origin Unavailable \(p. 82\)](#)
- [Transfer Encoding \(p. 82\)](#)
- [Dropped TCP Connections \(p. 83\)](#)

## Maximum File Size

The maximum size of a response body that CloudFront will return to the end user is 20 GB. This includes chunked transfer responses that don't specify the `Content-Length` header value.

## Caching

- Ensure that the origin server sets valid and accurate values for the `Date` and `Last-Modified` header fields.
- If requests from end users include the `If-Match` or `If-None-Match` request header fields, set the `ETag` response header field. If you do not specify an `ETag` value, CloudFront ignores subsequent `If-Match` or `If-None-Match` headers.

## Content Negotiation

The only acceptable value for the `Vary` header is `Accept-Encoding`. CloudFront ignores other values.

## Redirects

If you change the location of an object on the origin server, you can configure your web server to redirect requests to the new location. After you configure the redirect, the first time an end user submits a request for the object, CloudFront sends the request to the origin, and the origin responds with a redirect (for example, `302 Moved Temporarily`). CloudFront caches the redirect and returns it to the end user. CloudFront does not follow the redirect.

You can configure your web server to redirect requests to one of the following locations:

- The new URL of the object on the origin server. When the end user follows the redirect to the new URL, the end user bypasses CloudFront and goes straight to the origin. As a result, we recommend that you not redirect requests to the new URL of the object on the origin.
- The new CloudFront URL for the object. When the end user submits the request that contains the new CloudFront URL, CloudFront gets the object from the new location on your origin, caches it at the edge location, and returns the object to the end user. Subsequent requests for the object will be served by the edge location. This avoids the latency and load associated with viewers requesting the object from the origin. However, every new request for the object will incur charges for two requests to CloudFront.

## Origin Unavailable

If your origin server is unavailable and CloudFront gets a request for an object that is in the edge cache but that has expired (for example, because the period of time specified in the `Cache-Control max-age` directive has passed), CloudFront continues to serve the expired version of the object. For more information about object expiration, see [Specifying How Long Objects Stay in a CloudFront Edge Cache \(Object Expiration\)](#) (p. 65).

In some cases, an object that is seldom requested is evicted and is no longer available in the edge cache. CloudFront can't serve an object that has been evicted.

## Transfer Encoding

CloudFront supports only transfer encoding and requires the `Transfer-Encoding` header field value to be `"chunked"`.

## Dropped TCP Connections

If the TCP connection between CloudFront and your origin drops while your origin is returning an object to CloudFront, CloudFront behavior depends on whether your origin included a `Content-Length` header in the response:

- **Content-Length header:** CloudFront returns the object to the viewer as it gets the object from your origin. However, if the value of the `Content-Length` header doesn't match the size of the object, CloudFront doesn't cache the object.
- **No Content-Length header:** CloudFront returns the object to the viewer and caches it, but the object may not be complete. Without a `Content-Length` header, CloudFront cannot determine whether the TCP connection was dropped accidentally or on purpose.

## Supported HTTP Status Codes for Custom Origin Servers

If your custom origin server responds to a CloudFront request with any of the following status codes, CloudFront caches the code for five minutes and writes the results to the access logs.

204	No Content
305	Use Proxy
400	Bad Request
403	Forbidden
404	Not Found
405	Method Not Allowed
414	Request-URI Too Large
500	Internal Service Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Time-out

# Using a Signed URL to Serve Private Content

---

## Topics

- [Static Signed URLs vs. Dynamic Signed URLs \(p. 84\)](#)
- [Overview of Private Content \(p. 86\)](#)
- [How to Serve Private Content Using a Signed URL \(p. 88\)](#)
- [Signature Code, Examples, and Tools \(p. 110\)](#)

For many companies that distribute data via the Internet, it is important to restrict access to documents, business data, media streams, or content intended for users who have paid a fee. You can use CloudFront private distributions to restrict access to data in Amazon S3 buckets. This section describes how a private distribution is different from a public distribution, describes how to create a private distribution, and provides links to sample code you might find helpful when creating your signed URL.

## Important

You can use a signed URL to distribute content from a custom origin. However, for CloudFront to access your objects on your custom origin, the objects must remain publicly accessible. As a result, anyone who has the URL for an object on your custom origin can access the object without the protection provided by CloudFront signed URLs. If you use signed URLs with custom origins, do not give the URLs for the objects on your custom origin to your customers or to others outside your organization. For more information about origin servers for download distributions, see [Origins \(p. 40\)](#). For more information about origin servers for streaming distributions, see [Using an Amazon S3 Bucket as the Origin for a Streaming Distribution \(p. 48\)](#).

## Static Signed URLs vs. Dynamic Signed URLs

You can distribute private content with a static signed URL or a dynamic signed URL. You use a static signed URL when distributing private content to a known end user, such as distributing a business plan to an investor, or distributing training materials to employees. In this case, you create a signed URL and make the URL available to your end users as needed. You use a dynamic signed URL to distribute content on-the-fly to an end user for a limited purpose, such as distributing movie rentals or music downloads to customers on demand. In this case, your application generates the signed URL.



To integrate signed URL creation into your application for dynamic, on-the-fly signed URL generation, follow the procedures described in this section. However, to avoid coding, and yet distribute content to an end user for a limited purpose without dynamic signed URL creation, you can try creating a CloudFront private distribution using one of the third-party GUI tools listed in [GUI Tools for Signature Generation](#) (p. 125).

# Overview of Private Content

## Topics

- [Public vs. Private Content \(p. 86\)](#)
- [Two Parts to Serving Private Content \(p. 87\)](#)

This section describes the differences between public and private content, and explains the structure of a private content distribution.

## Public vs. Private Content

You can think of your content as being either *public* or *private* (i.e., restricted). The following table gives a general description of what each means.

Type of Content	General Description	General Implementation
Public	You want anyone to be able to access your objects	You set the Amazon S3 Access Control List (ACL) on the objects in your bucket to give <code>read</code> permission to everyone. This means that end users can access the objects through either CloudFront or Amazon S3.
Private	You want to restrict who can access your objects (for example, to control access to a paid download)	You set the Amazon S3 ACL on your objects so that only you and CloudFront have <code>read</code> permission for the objects. This means that end user access to the objects can <i>only</i> be through CloudFront. You also produce signed URLs for the end users you want to give access to.

A CloudFront private distribution is based on a *policy statement* that specifies any or all of the following constraints:

- A start date that specifies the date and time the signed URL will be valid
- An end date and time after which the signed URL will not be valid
- An IP address or range of IP addresses from which the signed URL can be used

## Note

The two initial API versions for CloudFront (2008-06-30 and 2009-04-02) require your content to be public. As of version 2009-09-09 you can serve public content, private content, or both.

## How Private Distributions Are Different from SSL and HTTPS

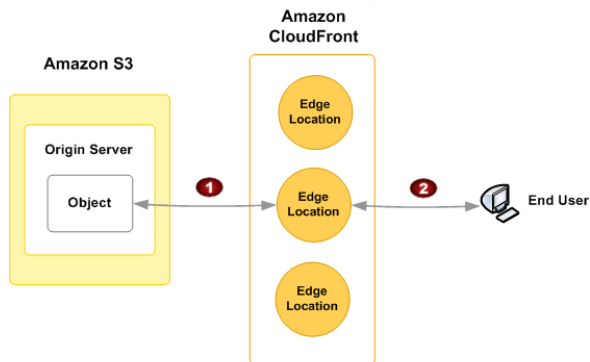
The security provided by a CloudFront private distribution is different from SSL and HTTPS, which encrypt the data that is transferred over a network connection. A CloudFront private distribution encodes request parameters that are appended to the base URL. The encoded parameters contain a *policy statement* (with the restrictions described in [Public vs. Private Content \(p. 86\)](#)) and a *signature*, which authenticates that the policy was generated by a trusted signer and has not been tampered with. When authentication

succeeds, the encoded content of the policy statement authorizes the request for data in the distribution during the specified time period and from the specified IP address or addresses.

For an overview of the process of creating private content, see [Two Parts to Serving Private Content \(p. 87\)](#).

## Two Parts to Serving Private Content

The process of serving private content has two main parts described in the following diagram and table.



<b>1</b>	<p>Securing the content in your bucket so that end users <i>only</i> have access to the content through CloudFront.</p> <p>You remove all public access to the objects in your buckets, but still give CloudFront permission to fetch the objects. To do this, you create a <i>CloudFront origin access identity</i>, which is a virtual identity that you give <code>read</code> permission to. Granting that permission allows CloudFront to fetch content that is otherwise marked as private in your bucket.</p>
<b>2</b>	<p>Restricting end user access to cached content.</p> <p>You hand out special signed URLs to end users you want to give access to. The URLs contain restrictions limiting access to the private content.</p>

You can do both parts, or just part 1. If you just do part 1, you still use public URLs just as you do to serve public content. The URL remains publicly accessible, but it is accessible only through CloudFront (not through your Amazon S3 bucket) and you don't need to use a signed URL. If you do part 2, the URL is private and is not publicly accessible.

For information about how to create a private content distribution using a signed URL, see [How to Serve Private Content Using a Signed URL \(p. 88\)](#).

# How to Serve Private Content Using a Signed URL

## Topics

- [Private Content Process Overview](#) (p. 88)
- [Securing Your Content in Amazon S3](#) (p. 90)
- [Restricting End User Access](#) (p. 94)

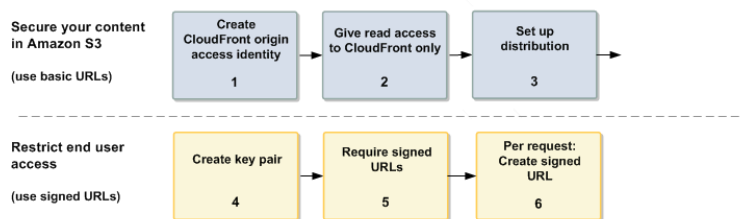
This section describes how to create a distribution you can use to distribute private content via a signed URL.

## Private Content Process Overview

The following figure and table describe the process for restricting user access. The process is divided into two sections corresponding to the two parts of the process (as described in [Two Parts to Serving Private Content](#) (p. 87)).

The first section, the top row with the first three tasks shaded in blue, covers the tasks required to secure your content in Amazon S3 (that is, to make your content accessible only through CloudFront).

The second section, with tasks 4 through 6, covers the additional tasks required to create signed URLs, which allow you to restrict access to the content to users who have a signed URL.



## Process for Serving Private Content

1	Use the CloudFront API to create a <i>CloudFront origin access identity</i> . For more information, see <a href="#">Overview of the CloudFront Origin Access Identity</a> (p. 90).
2	Use the Amazon S3 API or the Amazon S3 console of the AWS Management Console to update the ACL on your private objects. Give <code>read</code> permission to the CloudFront origin access identity by using its canonical user ID. Remove public-access permissions. For more information about setting the ACL, see <a href="#">Updating Amazon S3 Bucket Policies or ACLs on Your Private Content Buckets or Objects</a> (p. 92).
3	Set up a private content distribution or streaming distribution (either create a new distribution or update an existing one). For more information, see <a href="#">Creating a Private Content Distribution</a> (p. 92).
4	Use the <b>Accounts</b> link in the AWS Management Console to access the <b>Key Pairs</b> tab of the <b>Access Credentials</b> page. Create an RSA key pair and download the private key. You'll use this key to create a signed URL. For more information about creating your key pair, see <a href="#">Creating a Key Pair</a> (p. 94).

5	Update your private content distribution or streaming distribution to specify that the distribution's URLs must be signed, and the accounts that can sign them. For more information, see <a href="#">Requiring Signed URLs (p. 95)</a> .
6	Create a signed URL to give to the authorized end user. For more information, see <a href="#">Signature Code, Examples, and Tools (p. 110)</a> .

To get started with creating a private content distribution using a signed URL, see [Securing Your Content in Amazon S3 \(p. 90\)](#).

# Securing Your Content in Amazon S3

## Topics

- [Overview of the CloudFront Origin Access Identity \(p. 90\)](#)
- [Creating a CloudFront Origin Access Identity \(p. 91\)](#)
- [Creating a Private Content Distribution \(p. 92\)](#)
- [Updating Amazon S3 Bucket Policies or ACLs on Your Private Content Buckets or Objects \(p. 92\)](#)

To secure your content in Amazon S3, you create a CloudFront origin access identity and set the ACL on the objects or buckets in Amazon S3 to be accessible only by that identity. This section describes what an origin access identity is, shows you how to modify your distribution to include a CloudFront *origin access identity*, and how to authorize CloudFront access to data in Amazon S3.

## Overview of the CloudFront Origin Access Identity

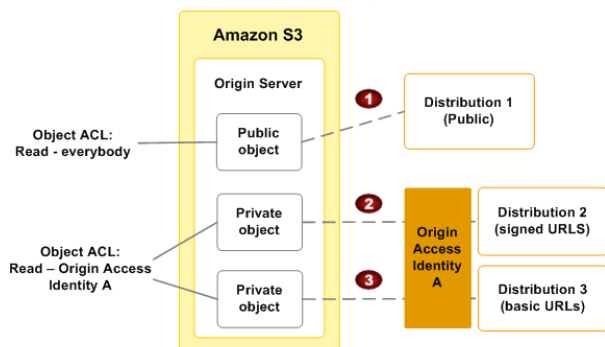
A CloudFront *origin access identity* is a virtual identity that allows CloudFront to fetch content from an Amazon S3 bucket. To use an origin access identity to secure your content in Amazon S3, you perform the following tasks:

- Create a CloudFront origin access identity for your AWS account. For more information, see [Creating a CloudFront Origin Access Identity \(p. 91\)](#).
- Add the origin access identity to your distribution. For download distributions, you can choose whether to add an origin access identity for each Amazon S3 origin. For streaming distributions, an origin access identity applies to the entire distribution. For more information, see [Creating a Private Content Distribution \(p. 92\)](#).
- Give the origin access identity `read` permission (or `read` and `download` permission) to objects in Amazon S3, and modify the bucket policy or any object ACLs as appropriate to ensure that the objects are not publicly available. For more information, see [Updating Amazon S3 Bucket Policies or ACLs on Your Private Content Buckets or Objects \(p. 92\)](#).

After you remove public access to the Amazon S3 bucket, the CloudFront distribution is now the only way to access objects in your bucket. Adding *signer* accounts to the distribution configuration allows access only to users who have signed URLs.

You can have up to 100 CloudFront origin access identities, and you can attach each to one or more distributions. One origin access identity is usually sufficient, even for multiple distributions.

The following example depicts three different distributions.



1	Distribution 1 is configured for public content. The object has an Amazon S3 ACL that grants everyone <code>read</code> permission. Anyone can access the contents of this distribution through Amazon S3 or through your CloudFront distribution.
2	Distribution 2 is configured to read private content with signed URLs. This distribution is attached to CloudFront origin access identity A. The object has an Amazon S3 ACL that grants <code>read</code> permission to the identity. The content in this distribution cannot be accessed by anyone who doesn't have the signed URL.
3	Distribution 3 is configured to read private content with public URLs. This distribution is also attached to CloudFront origin access identity A. The object has an Amazon S3 ACL that grants <code>read</code> permission to the identity. The content in this distribution is not private, but users can access it only through your CloudFront distribution (not through your Amazon S3 bucket).

## Creating a CloudFront Origin Access Identity

You can create a CloudFront origin access identity using a `POST` on the `2012-05-05/origin-access-identity/cloudfront` resource. You must provide a unique caller reference in the request, as you do when creating a distribution. You can optionally provide comments about the identity.

### Note

Currently, the AWS Management Console doesn't support creating an origin access identity or updating a distribution to serve private content.

### To create a CloudFront origin access identity for your distribution

1. Send a CloudFront control API request that is similar to the following example.

```
POST /2012-05-05/origin-access-identity/cloudfront HTTP/1.1
[Required headers]

<?xml version="1.0" encoding="UTF-8"?>
<CloudFrontOriginAccessIdentityConfig xmlns="http://cloudfront.amazonaws.com/doc/2012-05-05/">
  <CallerReference>20120229090000</CallerReference>
  <Comment>Your comments here</Comment>
</CloudFrontOriginAccessIdentityConfig>
```

2. You will receive a response that looks similar to the following example.

```
201 Created
Location: https://cloudfront.amazonaws.com/2012-05-05/origin-access-identity/cloudfront/E74FTE3AEXAMPLE
x-amz-request-id: request_id

<?xml version="1.0" encoding="UTF-8"?>
<CloudFrontOriginAccessIdentity xmlns="http://cloudfront.amazonaws.com/doc/2012-05-05/">
  <Id>E74FTE3AEXAMPLE</Id>
  <S3CanonicalUserId>
    cd13868f797c227fbea2830611a26fe0a21ba1b826ab4bed9b7771c9aEXAMPLE
  </S3CanonicalUserId>
```

```
<CloudFrontOriginAccessIdentityConfig>
  <CallerReference>20120229090000</CallerReference>
  <Comment>Your comments here</Comment>
</CloudFrontOriginAccessIdentityConfig>
</CloudFrontOriginAccessIdentity>
```

- Record the `Id` and the `S3CanonicalUserId` for the new CloudFront origin access identity. You will use these values later in the process. You use the `Id` to associate an origin access ID with your distribution, and the `S3CanonicalUserId` identifies CloudFront in the Amazon S3 ACL on the object. For more information about origin access ID or the Canonical User ID, go to [Actions on Origin Access Identities](#) in the *Amazon CloudFront API Reference*.

#### Note

The CloudFront API includes actions for creating and managing your CloudFront origin access identities. For more information, go to [Actions on Origin Access Identities](#) in the *Amazon CloudFront API Reference*.

Now that you have an origin access identity, you can create a distribution configured for private content. For more information, see [Creating a Private Content Distribution](#) (p. 92).

## Creating a Private Content Distribution

A distribution can serve either public or private content as specified by configuration values. To configure a distribution to serve private content, you use your AWS account, or a trusted AWS account you specify, to get a key pair. (If you already have an RSA key pair, you can upload the public key to AWS.) You then use the private key from the key pair to hash a policy statement; the result is a signature that you use to authenticate that the policy was generated by a trusted signer and has not been tampered with.

A private content distribution looks like a public content distribution, except that it has an `OriginAccessIdentity` element in the configuration. You must specify the value for the element using the following format: `origin-access-identity/cloudfront/ID`.

To create a private content distribution, use the CloudFront API to create a new distribution or update an existing distribution and include an `OriginAccessIdentity` element. See the applicable topic in the *Amazon CloudFront API Reference*:

- **Create a new download distribution:** [POST Distribution](#)
- **Update an existing download distribution:** [PUT Distribution Config](#)
- **Create a new streaming distribution:** [POST Streaming Distribution](#)
- **Update an existing streaming distribution:** [PUT Streaming Distribution Config](#)

Now that you have created a distribution configured for private content, you need to set the ACLs on your Amazon S3 private content objects. For more information, see [Updating Amazon S3 Bucket Policies or ACLs on Your Private Content Buckets or Objects](#) (p. 92).

## Updating Amazon S3 Bucket Policies or ACLs on Your Private Content Buckets or Objects

After you create a private content distribution, you must update Amazon S3 bucket policies or ACLs to grant the CloudFront origin access identity the permissions necessary to access to the private content in Amazon S3. Note the following:



- You may find it easier to update Amazon S3 bucket policies than ACLs because you can add objects to the bucket without updating permissions. However, ACLs give you more fine-grained control because you're granting permissions on each object.
- If you updated a public-content distribution to serve private content, modify the bucket policy or any object ACLs as appropriate to ensure that the objects are not publicly available.
- Both for bucket policies and for ACLs, when you specify the CloudFront entity to which you are granting access, use the `S3CanonicalUserId` element that was returned when you created a CloudFront origin access identity.

## Updating Amazon S3 Bucket Policies

Using either the AWS Management Console or the Amazon S3 API, change the Amazon S3 bucket policy to allow the CloudFront origin access identity to access objects in the bucket. For more information, go to [Using Bucket Policies](#) in the *Amazon Simple Storage Service Developer Guide*. For an example, see "Granting Permission, Using Canonical ID, to a CloudFront Origin Identify" in the topic [Example Cases for Amazon S3 Bucket Policies](#), also in the *Amazon Simple Storage Service Developer Guide*.

## Updating Amazon S3 ACLs

Using either the AWS Management Console or the Amazon S3 API, change the Amazon S3 ACL to give CloudFront `READ` permission on each object that the CloudFront distribution serves. For more information, go to [Using ACLs](#) in the *Amazon Simple Storage Service Developer Guide*.

You can also change the ACLs using code and one of the AWS SDKs. For an example, see the downloadable sample code in [Create a URL Signature Using C# and the .NET Framework \(p. 114\)](#).

## What's Next?

After you grant the CloudFront origin access identity the permissions necessary to access your Amazon S3 content, you may want to restrict end-user access to your distribution and create a signed URL. For more information, go to [Restricting End User Access \(p. 94\)](#).

## Restricting End User Access

### Topics

- [Creating a Key Pair \(p. 94\)](#)
- [Requiring Signed URLs \(p. 95\)](#)
- [Creating a Signed URL \(p. 97\)](#)

Restricting end user access involves creating a key pair, modifying your distribution to require signed URLs, and then creating the signed URL. This section describes these processes.

### Creating a Key Pair

*Signing a URL* is the process of creating an RSA digital signature using an RSA key and a policy statement. This section describes how to get the *key pair* consisting of a private key and a public key. AWS keeps the public key, and you keep the private key and use it to sign the URLs.

#### Important

The key pair is *not* an X.509 certificate and private key. It's an RSA key pair.

If you're an Amazon EC2 user, you probably already have at least one RSA key pair, which you use to connect to your EC2 instances through SSH or Windows Remote Desktop, but you can't reuse your EC2 key pairs with CloudFront because the key pair ID is not supplied. If you want to use your own key pair, see the procedure that follows for uploading your own public key to the AWS website.

### Using Your Own Key Pair

If you have a key pair that you want to use, you can upload the public key to AWS (you keep the private key). The public key must be an RSA key encoded in PEM format.

#### To upload your own public key

1. From the Amazon Web Services website at <http://aws.amazon.com>, point to **Your Account** and click **Security Credentials**.
2. Log in to your AWS account.  
The **Security Credentials** page is displayed.
3. In the **Access Credentials** section of the page, click the **Key Pairs** tab.
4. In the **Amazon CloudFront Key Pairs** area, click **Upload Your Own Key Pair**.
5. Follow the instructions presented to upload your public key.

### Using a Key Pair Generated by AWS

If you don't already have a key pair, you can have AWS generate a pair and automatically associate the public key with your AWS account.

#### To have AWS create a key pair for you

1. From the Amazon Web Services website at <http://aws.amazon.com>, point to **Your Account** and click **Security Credentials**.
2. Log in to your AWS account.  
The **Security Credentials** page is displayed.
3. In the **Access Credentials** section of the page, click the **Key Pairs** tab.

4. In the **Amazon CloudFront Key Pairs** area, click **Create a New Key Pair**.  
Your new public and private key are generated, along with an ID for the key pair. Amazon keeps the public key and gives you the private key.
5. From the dialog box, download your private key file to a local directory, and record the corresponding key pair ID.

You should keep your private key file secure. Make sure to set the permissions on the file so only you can read it. For a Linux/UNIX system, use `chmod 600`. To set the permission on a Windows system, right-click the file and set the file's security properties appropriately.

The next step is to configure your distribution to require signed URLs. For more information, see [Requiring Signed URLs \(p. 95\)](#).

## Requiring Signed URLs

You must configure your private content distribution to specify that URLs must be signed, and include the accounts that can sign them. Up to five AWS accounts other than your own can sign URLs for a single distribution. Each AWS account that you authorize must create and use its own key pair. For more information, see [Creating a Key Pair \(p. 94\)](#). A signed URL includes the signing key ID in the URL so that AWS can identify the signer account.

### To specify that URLs must be signed

- For a download distribution, add a `TrustedSigners` element to the applicable cache behaviors. For streaming distributions, add a `TrustedSigners` element to the distribution configuration.

### To specify who can sign URLs

1. If you want the AWS account that created the distribution to sign URLs, in the `TrustedSigners` element, add an `AwsAccountNumber` element that contains the value `self`.  
We don't assume that you do, so you must explicitly give permission to that account.
2. For each additional AWS account to which you want to give signing authority, add an `AwsAccountNumber` child element to the `TrustedSigners` element. There is a limit of five accounts, including `self`. Do not include dashes in the account numbers.

The AWS account number is displayed in the top right corner of the account owner's **Account Activity** page at <http://aws.amazon.com>.

For more information about specifying who can sign URLs, including examples, see the applicable documentation in the *Amazon CloudFront API Reference*:

- **Creating a download distribution:** [POST Distribution](#)
- **Updating a download distribution:** [PUT Distribution Config](#)
- **Creating a streaming distribution:** [POST Streaming Distribution](#)
- **Updating a streaming distribution:** [PUT Streaming Distribution Config](#)

Once you've specified trusted signers, you should verify that the signers are *active*. For a trusted signer to be active, both of the following must be true:

- The AWS account must have at least one *active* key pair. You can set a key pair to *inactive* when you rotate your keys. For more information, go to [Access Credential Rotation](#).
- CloudFront must be aware of the active key pair. After you create a key pair, there can be a short period of time before CloudFront is aware the key pair exists).

To determine which trusted signers are active trusted signers, get information about the distribution using the applicable CloudFront API action:

- **Download distributions:** [GET Distribution](#)
- **Streaming distributions:** [GET Streaming Distribution](#)

### Note

You must get the distribution, not just the distribution configuration.

The response includes an `ActiveTrustedSigners` element that lists the ID of each signer and the active key pairs associated with the trusted signer's AWS account. If a signer doesn't have an active key pair, CloudFront will not recognize that account as a signer.

The following example for a download distribution shows the status of the active trusted signers:

- The account that created the distribution (`self`) has an active key pair.
- The AWS account with ID 111122223333 has two active key pairs. (You can have a maximum of two active CloudFront key pairs at a time. For more information, see [Amazon CloudFront Key Pairs](#).)
- The third trusted signer (account ID 444455556666) doesn't currently have an active key pair (no `KeyPairId` appears for that signer), so the account can't create signed URLs.

```
200 OK
ETag: E2QWRUHEXAMPLE
x-amz-request-id: request_id

<Distribution xmlns="http://cloudfront.amazonaws.com/doc/2012-05-05/">
  <Id>EDFDVBD6EXAMPLE</Id>
  <Status>Deployed</Status>
  <LastModifiedTime>2012-05-19T19:37:58Z</LastModifiedTime>
  <InProgressInvalidationBatches>1</InProgressInvalidationBatches>
  <DomainName>d1111111abcdef8.cloudfront.net</DomainName>
  <ActiveTrustedSigners>
    <Quantity>3</Quantity>
    <Items>
      <Signer>
        <AwsAccountNumber>self</AwsAccountNumber>
        <KeyPairIds>
          <Quantity>1</Quantity>
          <Items>
            <KeyPairId>APKA9ONS7QCOWEXAMPLE</KeyPairId>
          </Items>
        </KeyPairIds>
      </Signer>
      <Signer>
        <AwsAccountNumber>111122223333</AwsAccountNumber>
        <KeyPairIds>
          <Quantity>2</Quantity>
          <KeyPairId>APKAI72T5DYBXEXAMPLE</KeyPairId>
          <KeyPairId>APKAU72D8DYNXEXAMPLE</KeyPairId>
        </KeyPairIds>
      </Signer>
      <Signer>
        <AwsAccountNumber>444455556666</AwsAccountNumber>
        <KeyPairIds>
          <Quantity>0</Quantity>
        </KeyPairIds>
      </Signer>
    </Items>
  </ActiveTrustedSigners>
</Distribution>
```

```
        </KeyPairIds>
      </Signer>
    </Items>
  </ActiveTrustedSigners>
  ...

```

The `ActiveTrustedSigners` element for a streaming distribution has the same syntax.

Next, you need to create your signed URL. For more information, see [Creating a Signed URL \(p. 97\)](#).

## Creating a Signed URL

### Topics

- [Overview of Signed URLs \(p. 97\)](#)
- [Creating a Policy Statement and a Signature \(p. 98\)](#)
- [Signed URL Examples \(p. 104\)](#)

This section gives an overview of signed URLs, describes how to create a policy statement and a signed URL, and provides some examples of how to create a signed URL.

### Overview of Signed URLs

A signed URL is composed of several parts. Following is an example of a CloudFront signed URL that uses a custom policy.

**1** <http://d1234EXAMPLE567.cloudfront.net/image.jpg?> **2** `Policy=eyJANCIAGlCEXAMPLEW1lbnQiOiBbeyANCiAGlCJSZXNvdXJjZSI6Imh0dHA6Ly9kemJlc3FtN3VuMW0wLmNsb3V`  
`kZnJvbnQubmV0L2RibW8ucGhwliwgDQogIENiAGlCQ29uZGI0aW9uLjlp7IA0KICAgIAGlCAGlkl`  
`wQWRkcmVzcyI6eyJBV1M6U291cmNISXAiOiIyMDcuMTcxLjE4MC4xMDEvMzlfSwNCiAGlCA`  
`glCAGlCJEYXRIR3JlYXRicldRoYW4iOnsiQVdTOkVwb2NoVGltZSI6MTI5Njg2MDE3Nn0sDQogI`  
`CAGlCAGlCAiRGF0ZUxlc3NUaGFuLjlp7IkFXUzpFcG9jaFRpbWUiOiJyOTY4NjAyMjZ9DQogICA`  
`glCB9IA0KICAgfV0gDQp9DQo=` **3** `&Signature=nitfHRCrtziwO2HwPFWw~yYDhUF5EwRunQ`  
`A-j19DzZrvDh6hQ73lDx~-ar3UocvvRQVw6EkC~GdpGQyyOSKQim-TxAnW7d8F5Kkai9HVx0F`  
`lu-5jcQb0UEmatEXAMPLE3ReXySpLSMj0yCd3ZAB4UcBCAqEijkytL6f3fVYNGQI=` **4** `&Key-`  
`Pair-Id=APLDHEXAMPLESQKAIQH3`

<b>1</b>	The CloudFront domain name with the file or media stream identifier. This is your base URL.
<b>2</b>	The policy statement request parameter. The policy statement was Base64-encoded, and several characters that are invalid in URL request parameters were replaced with valid characters. For more information, see <a href="#">Creating a Policy Statement and a Signature (p. 98)</a> .
<b>3</b>	The signature request parameter. The signature was Base64-encoded, and several characters that are invalid in URL request parameters were replaced with valid characters. For more information, see <a href="#">Creating a Policy Statement and a Signature (p. 98)</a> .
<b>4</b>	The Key-Pair-Id request parameter. This is the ID for the key pair that is associated with the account you are using to create the signature.

A canned policy, the simplest form of the signed URL, uses the signature, the ID, and an expiration date, but omits the encoded policy statement, because the only restriction is the expiration date. A custom policy requires the signature and policy request parameters, but not the expiration date, which, in the example above, is contained in the policy statement.

## Important

Your signed URL must not contain any whitespace. You might have to include escape characters in the string in application code.

The segments of a signed URL are described in the following examples.

The first segment is the CloudFront distribution domain name and the file to be retrieved, which in this case is `demo.txt`. The question mark (?) indicates that request parameters will follow.

```
http://dekrsgm7un9m0.cloudfront.net/demo.txt?
```

The following Base64 encoded string is the policy statement as a request parameter. Characters that are not valid in a request parameter have been replaced with valid characters. For more information, see [Creating a Policy Statement and a Signature](#) (p. 98).

```
Policy=eyJANCiAgICJTdGF0ZWllbnQiOiBbeyANCiAgICAgICJSZXNvdXJjZSI6Imh0dHA6Ly9kemJlc3FtN3VuMW0wLmNsb3VkZnJvbnQubmV0L2RlbW8ucGhwIiwgDQogICAgICAIQ29uZG10aW9uIjp7IA0KI  
CAGICAgICAgICklwQWRkcmVzcyI6eyJBVlM6U291cmNlSXAiOiIyMDcuMTcxLjE4MC4xMDEvMzIifSwNC  
iAgICAgICAgICJCYXRlR3JlYXRlc1RoYW4iOnsiQVdTOkVwb2NoVGltZSI6MTI5Njg2MDE3Nn0sDQogI  
CAGICAgICAgICRGF0ZUxlc3NUaGFuIjp7IkFXUzpFcG9jaFRpbWUiOjE5OTY4NjAyMjZ9DQogICAgICB9I  
A0KICAgfV0gdQp9DQo=
```

The next request parameter, indicated by the ampersand (&), is the Base64-encoded signature. As with the policy statement, characters that are not valid in a request parameter have been replaced with valid characters.

```
&Signature=nitfHRCrtziwO2HwPfWw~yYDhUF5EwRunQA-jl9DzZrvDh6hQ73lDx~-ar3UocvvRQVw6E  
kC~GdpGQyyOSKQim-TxAnW7d8F5Kkai9HVx0FIu-5jcQb0UEmatHw3FTxb3ReXySpLSMj0yCd3ZAB4Uc  
BCAqEijkytL6f3EXAMPLE=
```

The `Key-Pair-Id` request parameter is always required.

```
&Key-Pair-Id=APLDH2VGALRTSEXAMPLE
```

## Creating a Policy Statement and a Signature

### Topics

- [Canned Policy](#) (p. 100)
- [Custom Policy](#) (p. 102)

A policy statement specifies the restrictions on a signed URL. There are two types of policy statements: *canned* and *custom*. A canned policy statement is short and specifies only one condition: an end date after which the URL is invalid. A custom policy statement can include the start date, the end date, and the IP address or range of IP addresses for which the URL is valid. For both types of policy statement, the policy is defined in Java Script Object Notation (JSON) in UTF-8 format.

A signature is created by SHA1-hashing the policy statement and then encrypting the result by using RSA and the private key for your AWS account or for a trusted AWS account that you specify.

### Note

Sample signature code is available at [AWS Developer Resources Sample Code & Libraries](#), and in the section [Signature Code, Examples, and Tools](#) (p. 110). Additionally, CloudFront provides

a Perl script you can use to create an URL Signature. For more information about the Perl script, see [Create a URL Signature Using Perl \(p. 110\)](#).

### To create a signed URL

1. Create a policy statement. For more information, see the applicable section, [Canned Policy \(p. 100\)](#) or [Custom Policy \(p. 102\)](#).
2. If you are using a canned policy in the signed URL, skip to the next step.  
If you are using a custom policy, Base64-encode the policy statement, and replace invalid characters with valid characters to make the string URL-safe, as indicated in the following table. For an example, see [Signed URL Examples \(p. 104\)](#).

Invalid characters	Valid characters
+	-
=	_
/	~

3. Create a digital signature by SHA1-hashing the policy statement and RSA-encrypting the result using the private key for your AWS account or for a trusted AWS account that you specify. Then Base64-encode the result, and replace invalid characters with valid characters to make the string URL-safe, as indicated in the table in the previous step. For an example, see [Signed URL Examples \(p. 104\)](#).
4. Concatenate the CloudFront URL and the applicable parameters, depending on whether you are using a canned policy or a custom policy.

### Important

Your signed URL will not work if it cannot access the object origin. Ensure that you have granted read access to the private content to your CloudFront origin access identity. You do this by modifying the Amazon S3 ACL on each of the objects, not on the bucket. For more information, see [Updating Amazon S3 Bucket Policies or ACLs on Your Private Content Buckets or Objects \(p. 92\)](#).

The main part of this section covers how to correctly create the string that you sign. When the end user clicks the URL, the signature is verified and CloudFront evaluates the contents of the policy to determine if the end user is authorized access to the object specified by the URL. The following table describes the two types of policies you can use: *canned* or *custom*.

Type of Policy	Description
Canned	Lets you restrict access to a single URL based only on expiration time. You don't include the policy in the URL itself, so a canned policy results in a shorter URL. Because you don't include the policy in the URL, CloudFront constructs a <i>canned</i> policy based on information in the URL itself, and uses the canned policy to both validate the signature and determine if the end user has access to the content. For more information about using a canned policy, see <a href="#">Canned Policy (p. 100)</a> .

Type of Policy	Description
Custom	Lets you restrict access to one or more objects based on: end user IP address, a start time for access, and an expiration time for access. You must include the policy in the URL itself, so a custom policy results in a longer URL than a canned policy. CloudFront uses the policy statement to validate the signature and determine if the user has access to the content. For more information, see <a href="#">Custom Policy (p. 102)</a> .

### Canned Policy

A signed URL that uses a canned policy consists of a regular CloudFront URL, plus the three request parameters listed in the following table. You can use canned policies with HTTP and RTMP distributions.

Parameter	Description	Required
Expires	The expiration time of the URL, in epoch or UNIX time (number of seconds since January 1, 1970; e.g., 1258237200). Type: String	Yes
Signature	A URL-safe version of the signature. Type: String	Yes
Key-Pair-Id	The key ID of the signing private key. Type: String	Yes

The `Signature` value is an RSA-SHA1 digital signature of the following JSON policy, with the `RESOURCE` and `EXPIRES` values replaced as described in the table that follows the example.

### Example Canned Policy

```
{ "Statement": [ { "Resource": "RESOURCE", "Condition": { "DateLessThan": { "AWS:EpochTime": EXPIRES } } } ] }
```

### Important

For the signature you include in the URL to match the signature CloudFront constructs based on the canned policy, you must make sure the policy you construct looks exactly like the preceding policy. That is, you must use your own valid values for `RESOURCE` and `EXPIRES`, and you must remove any whitespace. You might have to include escape characters in the string in application code.

The following table describes the values to substitute in the policy.



Value in Policy	Substitute with...
RESOURCE (HTTP)	<p>The URL without the CloudFront request parameters <code>Expires</code>, <code>Signature</code>, and <code>Key-Pair-Id</code> parameters that will be created in the following steps. Retain any query parameters you've included in the URL. For example, if the full URL created by the signing process is:</p> <pre>http://d1111111abcdef8.cloudfront.net/download/horizon.jpg?large=yes&amp;license=yes &amp;Expires=1258237200&amp;Signature=<b>TBD</b>&amp;Key-Pair-Id=PK12345EXAMPLE</pre> <p>The value for <code>RESOURCE</code> is:</p> <pre>http://d1111111abcdef8.cloudfront.net/download/horizon.jpg?large=yes&amp;license=yes</pre> <p>If there are no request parameters, don't include the question mark in the value. Put the value inside quotation marks in the policy.</p>
RESOURCE (RTMP)	<p>With HTTP, a full URL uniquely describes an object. You can include the URL in the signature. The content of a streaming distribution, however, cannot always be described by a valid URL. In a streaming distribution, you only use the stream name to create a signature. For example, if your stream including the signature is:</p> <pre>example/mp3_name.mp3?Expires=1258237200&amp;Signature=TBD&amp;Key-Pair-Id=PK12345EXAMPLE</pre> <p>The value for <code>RESOURCE</code> is:</p> <pre>example/mp3_name</pre> <p>For streaming distributions, you do not include a prefix, such as <code>mp3:</code> or <code>mp4:</code>, for the resource name in the policy.</p> <p>Also, when referencing an MPEG file, you might have to omit the file extension for the URL enclosed in the signature. For example, you use <code>mp3_name</code> instead of <code>mp3_name.mp3</code>.</p>
EXPIRES	<p>The value of the <code>Expires</code> request parameter. Because the value is a number, which is specified in epoch seconds or UNIX time, you don't have to put quotation marks around it in the policy. For HTTP, we check the <code>Expires</code> value at the beginning of the HTTP request. For streaming content, CloudFront enforces the expiration only for play events. It is possible, for example, to play streaming content once, but when trying to replay it, the play fails because the UNIX time exceeded the value for <code>Expires</code>.</p>

To create the signature, you SHA1-hash the policy statement, RSA-encrypt the result using the private key for your AWS account or for a trusted AWS account that you specify, remove whitespace, Base64-encode that result, and replace characters that are invalid in a URL request parameter (`+`, `=`, `/`) with characters that are valid (`-`, `_`, and `~`, respectively). For an example, see [Signed URL Examples \(p. 104\)](#).

How you implement encryption of the signature depends on programming language and platform. This documentation provides examples in Perl, PHP, and C#. There are links to development resources for

Java. After you read the following sections that explain the details of policy statements for custom URL restrictions, look at the signature code examples in [Signature Code, Examples, and Tools \(p. 110\)](#).

### Important

The CloudFront process for creating a signature uses SHA1 and RSA. Amazon S3 and other AWS services use HMAC-SHA1.

Base64 encode the signature, and replace +, =, and / with -, \_, and ~, respectively, to make it URL safe before including it in the URL. (For more information, see [Signature Code, Examples, and Tools \(p. 110\)](#).)

For an example of a complete URL that uses a canned policy, see [Signed URL Examples \(p. 104\)](#).

### Custom Policy

A signed URL that uses a custom policy consists of a regular CloudFront URL, plus the three request parameters listed in the following table. Notice that no `Expires` parameter is used, as a canned policy would. A `Policy` parameter is required instead.

Parameter	Description	Required
<code>Policy</code>	A URL-safe version of the policy. The presence of the <code>Policy</code> parameter in the URL indicates a custom policy instead of a canned policy. Type: String	Yes
<code>Signature</code>	A URL-safe version of the signature Type: String	Yes
<code>Key-Pair-Id</code>	The key ID of the signing private key. Type: String	Yes

The policy you use to create the signature is a JSON document in UTF-8 format that specifies the resource and any conditions for accessing the resource. Use the policy format shown in the following example and the parameters listed in the following table.

The following example is a policy statement that allows access to the `game_download.zip` object if the end user's IP address is within the 192.0.2.0/24 IP address range, and if the end user's request for the object comes in before 11/14/2011 at 10:20 p.m., as specified in epoch time in seconds. There should be one newline character at the end of the custom policy statement.

### Example Custom Policy

```
{
  "Statement": [{
    "Resource": "http://d1111111abcdef8.cloudfront.net/game_download.zip",
    "Condition": {
      "IpAddress": { "AWS:SourceIp": "192.0.2.0/24" },
      "DateLessThan": { "AWS:EpochTime": 1258237200 }
    }
  }]
}
```

## Important

If you're familiar with Amazon S3 browser-based POSTs, the policy format you use there differs from the policy format you use here for CloudFront private objects.

If you're familiar with Amazon SQS access control, CloudFront uses the same format for its policies, but limits how you can use the syntax only for the following specific cases.

The following table describes the parameters you can specify in custom policy.

Parameter	Description	Required
Resource (HTTP)	<p>The URL to the cached object itself. This is the <i>CloudFront URL</i> (using the CloudFront domain name), not the URL to the object in the Amazon S3 bucket.</p> <p>The value must include the <code>http://</code>, and must match the resource specified in the URL. You can use multi-character match wild cards (*) or a single-character match wild card (?) anywhere in the string. For example, the value could be <code>http://d11111labdef8.cloudfront.net/*game_download.zip*</code>. This would include (for example) <code>example_game_download.zip?license=yes</code>.</p> <p>Omitting this parameter gives the end user access to <i>all</i> the objects belonging to any distribution associated with the key pair used to sign the URL.</p>	Optional
Resource (RTMP)	<p>When you use streaming content as your resource, use the stream name in the policy, for example, <code>example/mp3_name.mp3?Expires=1258237200&amp;Signature=TBD&amp;Key-Pair-Id=PK12345EXAMPLE</code>.</p> <p>When referencing an MPEG file, you might need to omit the file extension for the URL enclosed in the signature. For example, you use <code>mp3_name</code> instead of <code>mp3_name.mp3</code>.</p>	Optional
DateLessThan	<p>This is the only required parameter. It specifies an expiration date and time for the URL, using the format <code>"AWS:EpochTime":value</code> in seconds, and with no quotation marks. We require this value to prevent permanent access to any private content.</p>	Required
DateGreaterThan	<p>Specifies an optional start date and time for the URL, using the format <code>"AWS:EpochTime":value</code> in seconds, and with no quotation marks.</p>	Optional
IpAddress	<p>Specifies the IP address of the client making the GET request, using the format <code>"AWS:SourceIp":value</code>.</p> <p>It must be in standard CIDR format (for example, <code>10.52.176.0/24</code>). For more information, go to <a href="#">RFC 4632</a>. You can specify only a single value for the condition. For example, you can't set the policy to allow access if the client's IP address is in one of two different ranges.</p> <p>To allow access to all IP addresses, omit this parameter.</p>	Optional

The parameter names must be specified in the policy exactly as shown in the preceding table (no abbreviations like `dateLt` for `DateLessThan` are accepted). The order of the parameters in the policy

doesn't matter. Specify the conditions (`DateLessthan`, `DateGreaterthan`, and `IpAddress`) as part of the `Condition` section in the policy as shown in the examples in [Signed URL Examples \(p. 104\)](#).

The square brackets that enclose the statement's contents, as shown in the preceding example policy, are required for the policy to be valid.

You calculate the `Signature` request parameter using an RSA-SHA1 digital signature of the policy statement (the policy must be in UTF-8 format before signing).

#### Note

Signature encryption depends on platform and code language options. For more information about signature generation in various code languages, see [Signature Code, Examples, and Tools \(p. 110\)](#)

Base64 encode the signature, and replace `+`, `=`, and `/` with `-`, `_`, and `~`, respectively, to make it URL safe. For more information, see [Signature Code, Examples, and Tools \(p. 110\)](#).

## Signed URL Examples

This section shows example signatures based on the non-working credentials in the following table. You can download the example credentials zipped in the [CloudFront\\_PrivateContent\\_SignatureExamples.zip](#) file.

Credential	Value
Public Key	<pre> ----- BEGIN PUBLIC KEY ----- MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDA7ki9gI/lRygIoOjVlyymgx6FYFlzJ+z1ATMaLo57nL57AavWhb68HYY8EA0GJU9xQdMVaHBogF3eiCWYXSUZCWM/+M5+ZcdQraRRScucmn6g4EvY2K4W2pxbqH8vmUikPxir41EeBPLjMOzKvbzzQy9e/zziQVREKSp/7y1myEXAMPLE ----- END PUBLIC KEY ----- </pre>
Private Key	<pre> ----- BEGIN RSA PRIVATE KEY ----- MIICXQIBAAKBgQDA7ki9gI/lRygIoOjVlyymgx6FYFlzJ+z1ATMaLo57nL57AavWhb68HYY8EA0GJU9xQdMVaHBogF3eiCWYXSUZCWM/+M5+ZcdQraRRScucmn6g4EvY2K4W2pxbqH8vmUikPxir41EeBPLjMOzKvbzzQy9e/zziQVREKSp/7y1myWIDAQABAoGABc7mp7XYHynuPZxChjWNJZiQ+A73gm0ASDv6At7F8Vi9r0xUlQe/v0AQs3ycN8QlyR4XMbzMLYk3yJxFDXo4ZKQtOGzLGteCU2srANiLv26/imXA8FVidZftTatLviWQZBVPteYIA69ATUYPEq0a5u5wjGyU0ij9OWyuy01mbPkCQQDluYoNpPOekQ0ZWrPgJ5rxc8f6zG37ZVoDBiexqtVShIF5W3xYuWhW5kYb0hliYfkq15cS7t9m95h3lQJf/xI/AkEA1v9l/WN1a1N3rOK4VGoCokx7kR2SyTMSbZgF9IWJNOugR/WZw7HTnjipO3c9dy1Ms9pUKwUF46d7049ck8HwdQJARgrSKuLWXMyBH+/l1Dx/I4tXuAJIrlPyo+VmiOc7b5NzHptkSHEPFR9s1OK0Vqjknc1qCJ3Ig86OMetEFBzjZQJBAKYz470hcPkaGk7tKYAgP48FvxRsnzeooptURW5E+M+PQ2W9iDPPOX9739+Xi02hGEWFBOIGbQoTRFdE4VVcPK0CQQCeS84lODlC0Y2BZv2JxW3Osv/WkUQ4ds1fAQ11T3037uwwr7XTroMv8dIFQIPreOphRKmd/SbJzbiKfEXAMPLE ----- END RSA PRIVATE KEY ----- </pre>
Key-Pair-Id	PK12345EXAMPLE

Using the OpenSSL package, you can calculate the request parameters as shown in the following examples. For information about OpenSSL, go to <http://www.openssl.org>.

The following command creates the URL-safe Policy value.

```
% cat policy | openssl base64 | tr '+=/' '-_~'
```

The following command creates the URL-safe Signature value.

```
% cat policy | openssl sha1 -sign private-key.pem | openssl base64 | tr '+=/' '-_~'
```

### Note

You must remove whitespace from the resulting Base64 encoding.

For code examples that demonstrate creating a signature in several programming languages see [Signature Code, Examples, and Tools \(p. 110\)](#)

### Example Canned Policy

The following canned policy example gives any user with the signed URL access to `http://d604721fxaaqy9.cloudfront.net/horizon.jpg` before Mon, 14 Nov 2011 22:20:00 GMT.

The original URL is the CloudFront URL and request parameters.

```
http://d604721fxaaqy9.cloudfront.net/horizon.jpg?large=yes&license=yes
```

The policy statement that will be hashed and encrypted into the signature uses the original URL and an expiration time in epoch/UNIX seconds. If you copy and paste this example, remove any whitespace, and replace the URL and expiration time with your own values.

```
{ "Statement": [{ "Resource": "http://d604721fxaaqy9.cloudfront.net/horizon.jpg?large=yes&license=yes", "Condition": { "DateLessThan": { "AWS:EpochTime": 1258237200 } } } ] }
```

The signature is the result of SHA1 hashing, RSA encryption, and Base64 encoding of the result. You must also replace +, =, and / with -, \_, and ~, respectively, to make the value URL safe. For more information about this process, see [Signature Code, Examples, and Tools \(p. 110\)](#).

```
Signature = Nql641NHEUkUaXQHZINK1FZ~SYeUSoBJMxjdgqrzIdzV2gyEXPDNv0pYdWJkflDKJ3xIu7lbwRpSkG98NBlgPi4ZJpRRnVX4kXAJK6tdNx6FucDB7OVqzcckxHsGFd8VCG1BkC-Afh9~lOCMIYHIaiOB6~5jt9w2EOwiEXAMPLE_
```

Following is the full URL for the authorized user.

```
http://d604721fxaaqy9.cloudfront.net/horizon.jpg?large=yes&license=yes&Expires=1258237200&Signature=Nql641NHEUkUaXQHZINK1FZ~SYeUSoBJMxjdgqrzIdzV2gyEXPDNv0pYdWJkflDKJ3xIu7lbwRpSkG98NBlgPi4ZJpRRnVX4kXAJK6tdNx6FucDB7OVqzcckxHsGFd8VCG1BkC-Afh9~lOCMIYHIaiOB6~5jt9w2EOwiEXAMPLE_&Key-Pair-Id=PK12345EXAMPLE
```

The objective of the following custom policy example is to grant the network 145.168.143.0/24 access to all the objects in the `training` directory before Mon, 14 Nov 2011 22:20:00 GMT.

<http://d604721fxaaqy9.cloudfront.net/training/orientation.avi>

```
{
  "Statement": [{
    "Resource": "http://d604721fxaagy9.cloudfront.net/training/*",
    "Condition": {
      "IpAddress": { "AWS:SourceIp": "145.168.143.0/24" },
      "DateLessThan": { "AWS:EpochTime": 1258237200 }
    }
  }]
}
```

`Policy=eyJAKICAgIlN0YXRlbWVudCI6IjF7IAogICAgICAiUmVzb3VyY2UiOiJodHRwOi8vZDYwNDcyMWZ4YWFXeTkuY2xvdWRmcm9udC5uZXQvdHJaW5pbmcvcKiIsIAogICAgICAiQz29uZGt0aW9uIjp7IAogICAgICAgICAiSXBBBGRyZXNzIjp7IkFXUzp7b3VyY2VJcCI6IjEjeONS4xNjguMTQzMjQlLjAvMjQifSwgCiAgICAgICAgICAgICJEYXRLTGVCZclRoYW4iOnsiQVdTOkVwb2NoVGltZSI6MTI1ODIzNmIwMH0gICAgICAgICAgICAgfSAKICAgfEXAMPLE`

Signature=cPfTRKvUfYNYmxek6ZNs6vgKEZP6G3Cb4cyVt~FjqbHOnMdxdt7eT6pYmhHYzuDsFH4Jps  
ctke2Ux6PCXcKxUcTiM8SO4b29~1QvhM1~CIojki3Hd3~Unxjw7CpolqRjtvrinW0DPZBZYHFZtiZX  
saPt87yBP9GwnTQoEXAMPLE\_

[illegible]

For code examples that demonstrate creating a signature in several programming languages see [Signature Code, Examples, and Tools](#) (p. 110)

The objective of the following custom policy is to grant the IP addresses 216.98.35.1/32 access to all the objects belonging to any distribution that the specified key pair ID is associated with. The objects are to be available only between Sat, 30 Apr 2011 06:43:10 GMT and Sun, 16 Oct 2011 06:31:56 GMT.

<http://d841721fxaagy9.cloudfront.net/downloads/pictures.tgz>

```
{
  "Statement": [{
    "Resource": "http://*",
    "Condition": {
      "IpAddress": { "AWS:SourceIp": "216.98.35.1/32" },
      "DateGreaterThan": { "AWS:EpochTime": 1241073790 },
      "DateLessThan": { "AWS:EpochTime": 1255674716 }
    }
  }]
}
```

[illegible]

Signature=rc~5Qbbm8EJXjTQ6Cn0LAXr72g1DOPrTmdtfbWVVgQNw0q~KHUAmB  
a2Zv1Wjj8dDET4XSL~Myh44CLQdu4dOH~N9huH7QfPSR~O4tIOS1WWcP~2JmtVPoQyLlEc8YHRCuN3nVN  
ZJ0m4EZcXXNAS~0x6Zco2SYx~hywEXAMPLE\_

<http://d84l721fxaagy9.cloudfront.net/downloads/pictures.tgz?Policy=eyJAKICAgILN0YXRlbWVudCI6IjE7IAogICAgICAiUmVzb3VyZyUOiJodHRwOi8vKiIsIAogICAgICAiQz9uZGl0aw9uIjp7IAogICAgICAgICAiSXBZZGRyZXNZIjp7IkFXUZpbTb3VyZ2VJcCIE6IjIxNi45OC4zNS4xLzM5In0sCiAgICAgICAgICAgICJEYXRlR3JlYXRsclRoYW4iOnsiQVdTOKkVwb2NoVGltZSI6MTIOMTA3Mzc5MH0sCiAgICAgICAgICAgICJEYXRlTGVCZClRoYW4iOnsiQVdTOKkVwb2NoVGltZSI6MTIINTYzNDcxNm0KICAgICAgfSAKICAgfEXAMPLE&Signature=rc~5Qbbm8EJxjUTQ6CnOLAxR72gIDOPrTmdtf>



```
bWVVgQNw0q~KHUAmBa2Zv1Wjj8dDET4XSL~Myh44CLQdu4dOH~N9huH7QfPSR~O4tIOSlWWcP~2JmtV  
PoQyLlEc8YHRCuN3nVNZJ0m4EZcXXNAS-0x6Zco2SYx~hywEXAMPLE_&Key-Pair-  
Id=PK12345EXAMPLE
```

For code examples that demonstrate creating a signature in several programming languages see [Signature Code, Examples, and Tools](#) (p. 110).



The following example uses jwplayer with CloudFront.

```
<script type='text/javascript'>
  var sol = new SWFObject
    ('http://d841721fxaaqy9.cloudfront.net/player/player.swf',
    'mpl', '640', '360', '9');
  sol.addParam('allowfullscreen','true');
  sol.addParam('allowscriptaccess','always');
  sol.addParam('wmode','opaque');
  sol.addVariable('streamer','rtmp://s33r3xe4ayhhis.cloudfront.net/cfx/st');
  sol.addVariable("file","mp4:example/video.mp4%3FPolicy%3DewogICJTdGF0ZWllbnQiOlt7CiAgICAgICJSZXNvdXJzZSI6ImRyciIsCiAgICAgICJDb25kaXRpb24iOnsKICAgICAgICAiSXBZGRyZXNzIjp7IkFXUzpTb3VyY2VJcCI6IjAuMC4wLjAvMCJ9LAogICAgICAgICJEYXRlTGZvcz1RoYW4iOnsiQVdTOKVwb2NoVGltZSI6MjE0NTkxNjgwMH0KICAgICAgfQogICAgEXAMPLE_%26Signature%3DewtHqEXK~68tsZt-eOfnZKGWtf2aJlbKhXkK5SSiVqcG9pieCRV3xTEPtC290zeXlsDvRycOM2WK0cXzcyYZhpl9tv2796ihHiCTAwIHQ8yP17Af4nWtOLIZHoH6wkr3tUlcQHs8Rld-g-SlZGjNBXr~J2MbaJzm8i6EXAMPLE_%26Key-Pair-Id%3DPK12345EXAMPLEsol.write('flv');
</script>
```

When you retrieve a stream to play from within an Adobe Flash .swf file, do not URL-encode the stream name, for example:

```
mp4:example/video.mp4?Policy=ewogICJTdGF0ZWllbnQiOlt7CiAgICAgICJSZXNvdXJzZSI6ImRyciIsCiAgICAgICJDb25kaXRpb24iOnsKICAgICAgICAiSXBZGRyZXNzIjp7IkFXUzpTb3VyY2VJcCI6IjAuMC4wLjAvMCJ9LAogICAgICAgICJEYXRlTGZvcz1RoYW4iOnsiQVdTOKVwb2NoVGltZSI6MjE0NTkxNjgwMH0KICAgICAgfQogICAgEXAMPLE_&Signature=ewtHqEXK~68tsZt-eOfnZKGWtf2aJlbKhXkK5SSiVqcG9pieCRV3xTEPtC290zeXlsDvRycOM2WK0cXzcyYZhpl9tv2796ihHiCTAwIHQ8yP17Af4nWtOLIZHoH6wkr3tUlcQHs8Rld-g-SlZGjNBXr~J2MbaJzm8i6EXAMPLE_&Key-Pair-Id=PK12345EXAMPLE
```

See the comments in the Perl source code for more information about the command line switches and features of this tool.

See also

- [Create a URL Signature Using PHP \(p. 111\)](#)
- [Create a URL Signature Using C# and the .NET Framework \(p. 114\)](#)
- [Create a URL Signature Using Java \(p. 122\)](#)
- [GUI Tools for Signature Generation \(p. 125\)](#)

## Create a URL Signature Using PHP

Any web server that runs PHP can use the PHP demo code to create policy statements and signatures for CloudFront streaming private distributions. The sample creates a functioning web page with signed URL links that play a video stream using CloudFront streaming. To get the sample, download [Signature Code for Video Streaming in PHP](#).

## Note

Creating a URL signature is just one part of the process of serving private content using a signed URL. For more information about the entire process, see [How to Serve Private Content Using a Signed URL \(p. 88\)](#).

In the following code segment, the function `rsa_shal_sign` hashes the policy and encrypts the result. The arguments required are a policy statement, an out parameter to contain the signature, and the private key for your AWS account or for a trusted AWS account that you specify. Next, the `url_safe_base64_encode` function creates a URL-safe version of the signature.

## Example RSA SHA1 Encryption in PHP

```
function rsa_shal_sign($policy, $private_key_filename) {
    $signature = "";

    // load the private key
    $fp = fopen($private_key_filename, "r");
    $priv_key = fread($fp, 8192);
    fclose($fp);
    $pkeyid = openssl_get_privatekey($priv_key);

    // compute signature
    openssl_sign($policy, $signature, $pkeyid);

    // free the key from memory
    openssl_free_key($pkeyid);

    return $signature;
}

function url_safe_base64_encode($value) {
    $encoded = base64_encode($value);
    // replace unsafe characters +, = and / with
    // the safe characters -, _ and ~
    return str_replace(
        array('+', '=', '/'),
        array('-', '_', '~'),
        $encoded);
}
```

The following code constructs a *canned* policy statement needed for creating the signature. For more information about canned policies, see [Canned Policy \(p. 100\)](#).

### Example Canned Signing Function in PHP

```
function get_canned_policy_stream_name($video_path, $private_key_filename,
$key_pair_id, $expires) {
    // this policy is well known by CloudFront, but you still need to sign it,

    // since it contains your parameters
    $canned_policy = '{"Statement":[{"Resource":"' . $video_path . '","Condition":{"DateLessThan":{"AWS:EpochTime":"' . $expires . '}}}]}' ;
    // the policy contains characters that cannot be part of a URL,
    // so we Base64 encode it
    $encoded_policy = url_safe_base64_encode($canned_policy);
    // sign the original policy, not the encoded version
    $signature = rsa_shal_sign($canned_policy, $private_key_filename);
    // make the signature safe to be included in a url
    $encoded_signature = url_safe_base64_encode($signature);

    // combine the above into a stream name
    $stream_name = create_stream_name($video_path, null, $encoded_signature,
$key_pair_id, $expires);
    // url-encode the query string characters to work around a flash player bug

    return encode_query_params($stream_name);
}
```

The following code constructs a *custom* policy statement needed for creating the signature. For more information about custom policies, see [Canned Policy \(p. 100\)](#).

### Example Custom Signing Function in PHP

```
function get_custom_policy_stream_name($video_path, $private_key_filename,
$key_pair_id, $policy) {
    // the policy contains characters that cannot be part of a URL, so we Base64
    encode it
    $encoded_policy = url_safe_base64_encode($policy);
    // sign the original policy, not the encoded version
    $signature = rsa_shal_sign($policy, $private_key_filename);
    // make the signature safe to be included in a url
    $encoded_signature = url_safe_base64_encode($signature);

    // combine the above into a stream name
    $stream_name = create_stream_name($video_path, $encoded_policy, $encoded_sig
nature, $key_pair_id, null);
    // url-encode the query string characters to work around a flash player bug

    return encode_query_params($stream_name);

}
```

For more information about OpenSSL implementation of RSA encryption, see [The Open Source Toolkit for SSL/TLS](#).

See also

- [Create a URL Signature Using Perl \(p. 110\)](#)
- [Create a URL Signature Using C# and the .NET Framework \(p. 114\)](#)
- [Create a URL Signature Using Java \(p. 122\)](#)

- [GUI Tools for Signature Generation \(p. 125\)](#)

## Create a URL Signature Using C# and the .NET Framework

The C# examples in this section implement a sample application that demonstrates how to create the signatures for CloudFront private distributions using canned and custom policy statements. The samples includes utility functions based on the [AWS .NET SDK](#) that can be useful in .NET applications.

### Note

Creating a URL signature is just one part of the process of serving private content using a signed URL. For more information about the entire process, see [How to Serve Private Content Using a Signed URL \(p. 88\)](#).

To download the code, go to [Signature Code in C#](#).

To use the RSA keys provided by [AWS Account/Security](#) in the .NET framework, you must convert the AWS-supplied .pem files to the XML format that the .NET framework uses. The OpenSSL Public and Private Key Parser available from [.NET 2.0 OpenSSL Public and Private Key Parser](#) will do the conversion.

After conversion, the RSA private key file is in the following format:

### Example RSA Private Key in the XML .NET Framework Format

```
<RSAKeyValue>
  <Modulus>
    wO5IvYCP5UcoCKDo1dcspoMehWBZcyfs9QEzGi6Oe5y+ewGr1oW+vB2GPB
    ANBiVPcUHTFWHwaIBd3oglmF0lGQljP/jOfmXHUK2kUUnLnJp+oOBL2Ni
    uFtqcW6h/L5lIpD8Yq+NRHg
    Ty4zDsyr2880MvXv88yEFURckqEXAMPLE=
  </Modulus>
  <Exponent>AQAB</Exponent>
  <P>
    5bmKDaTz
    npENGvqz4Cea8XPH+sxt+2VaAwYnsarVUoS
    BeVt8WLloVuZGG9IZYmH5KteXEu7fZveYd9UEXAMPLE==
  </P>
  <Q>
    1v9l/WN1a1n3rOK4VGoCokx7kr2SyTMSbZgF9IWJNOugR/WZw7HTnjip03c9dy1Ms9pUKwUF4
    6d7049EXAMPLE==
  </Q>
  <DP>
    RgrSKuLWXMyBH+/l1Dx/I4tXuAJIrlPyo+VmiOc7b5NzHptkSHEPfR9s1
    OK0VqjknclqCJ3Ig86OMEtEXAMPLE==
  </DP>
  <DQ>
    pjPjvSFw+RoaTu0pgCA/jwW/FGyfn6iim1RFbkt4
    z49DZb2IM885f3vf35eLTaEYRYUHQgZtChNEV0TEXAMPLE==
  </DQ>
  <InverseQ>
    nkvoJTG5QtGNgWb9i
    cVtzrL/1pFEOHbJXwEJdU99N+7sMK+1066DL/HSBUCD63qD4USpnf0myc24in0EXAMPLE==</InverseQ>
  <D>
    Bc7mp7XYHynuPZxChjWNJZiQ+A73gm0ASDv6At7F8Vi9r0xU1Qe/v0AQS3ycN8QlyR4XMbzMLYk
    3yjxFDXo4ZKQtOGzLGteCU2srANiLv26/imXA8FVidZftTAtLviWQZB
    VPTeYIA69ATUYPEq0a5u5wjGy
    UOi j9OWyuEXAMPLE=
  </D>
</RSAKeyValue>
```

The following C# code creates a signed URL that uses a canned policy by:

- Creating a policy statement.
- Hashing the policy statement using SHA1, and encrypting the result using RSA and the private key for your AWS account or for a trusted AWS account that you specify.
- Base64-encoding the hashed policy statement and replacing special characters to make the string safe to use as a URL request parameter.
- Concatenating the applicable values.

For the complete implementation, see the sample at [Signature Code in C#](#).

### Example Canned Policy Signing Method in C#

```
public static string ToUrlSafeBase64String(byte[] bytes)
{
    return System.Convert.ToBase64String(bytes)
        .Replace('+', '-')
        .Replace('=', '_')
        .Replace('/', '~');
}

public static string CreateCannedPrivateURL(string urlString,
    string durationUnits, string durationNumber, string pathToPolicyStmnt,
    string pathToPrivateKey, string privateKeyId)
{
    // args[] 0-thisMethod, 1-resourceUrl, 2-seconds-minutes-hours-days
    // to expiration, 3-numberOfPreviousUnits, 4-pathToPolicyStmnt,
    // 5-pathToPrivateKey, 6-PrivateKeyId

    TimeSpan timeSpanInterval = GetDuration(durationUnits, durationNumber);

    // Create the policy statement.
    string strPolicy = CreatePolicyStatement(pathToPolicyStmnt,
        urlString,
        DateTime.Now,
        DateTime.Now.Add(timeSpanInterval),
        "0.0.0.0/0");
    if ("Error!" == strPolicy) return "Invalid time frame." +
        "Start time cannot be greater than end time.";

    // Copy the expiration time defined by policy statement.
    string strExpiration = CopyExpirationTimeFromPolicy(strPolicy);

    // Read the policy into a byte buffer.
    byte[] bufferPolicy = Encoding.ASCII.GetBytes(strPolicy);

    // Initialize the SHA1CryptoServiceProvider object and hash the policy data.
    using (SHA1CryptoServiceProvider
        cryptoSHA1 = new SHA1CryptoServiceProvider())
    {
        bufferPolicy = cryptoSHA1.ComputeHash(bufferPolicy);

        // Initialize the RSACryptoServiceProvider object.
        RSACryptoServiceProvider providerRSA = new RSACryptoServiceProvider();

        XmlDocument xmlPrivateKey = new XmlDocument();

        // Load PrivateKey.xml, which you created by converting your
        // .pem file to the XML format that the .NET framework uses.
        // Several tools are available. We used
        // .NET 2.0 OpenSSL Public and Private Key Parser,
        // http://www.jensign.com/opensslkey/opensslkey.cs.
        xmlPrivateKey.Load(pathToPrivateKey);

        // Format the RSACryptoServiceProvider providerRSA and
        // create the signature.
        providerRSA.FromXmlString(xmlPrivateKey.InnerXml);
    }
}
```



```
RSAPKCS1SignatureFormatter rsaFormatter =
    new RSAPKCS1SignatureFormatter(providerRSA);
rsaFormatter.SetHashAlgorithm("SHA1");
byte[] signedPolicyHash = rsaFormatter.CreateSignature(bufferPolicy);

// Convert the signed policy to URL-safe Base64 encoding and
// replace unsafe characters + = / with the safe characters - _ ~
string strSignedPolicy = ToUrlSafeBase64String(signedPolicyHash);

// Concatenate the URL, the timestamp, the signature,
// and the key pair ID to form the signed URL.
return urlString +
    "?Expires=" +
    strExpiration +
    "&Signature=" +
    strSignedPolicy +
    "&Key-Pair-Id=" +
    privateKeyId;
}
```

The following C# code creates a signed URL that uses a custom policy by:

- Creating a policy statement.
- Base64-encoding the policy statement and replacing special characters to make the string safe to use as a URL request parameter.
- Hashing the policy statement using SHA1, and encrypting the result using RSA and the private key for your AWS account or for a trusted AWS account that you specify.
- Base64-encoding the hashed policy statement and replacing special characters to make the string safe to use as a URL request parameter.
- Concatenating the applicable values.

For the complete implementation, see the sample at [Signature Code in C#](#).

### Example Custom Policy Signing Method in C#

```
public static string ToUrlSafeBase64String(byte[] bytes)
{
    return System.Convert.ToBase64String(bytes)
        .Replace('+', '-')
        .Replace('=', '_')
        .Replace('/', '~');
}

public static string CreateCustomPrivateURL(string urlString,
    string durationUnits, string durationNumber, string startIntervalFromNow,

    string ipaddress, string pathToPolicyStmnt, string pathToPrivateKey,
    string PrivateKeyId)
{
    // args[] 0-thisMethod, 1-resourceUrl, 2-seconds-minutes-hours-days
    // to expiration, 3-numberOfPreviousUnits, 4-starttimeFromNow,
    // 5-ip_address, 6-pathToPolicyStmnt, 7-pathToPrivateKey, 8-privateKeyId

    TimeSpan timeSpanInterval = GetDuration(durationUnits, durationNumber);
    TimeSpan timeSpanToStart = GetDurationByUnits(durationUnits,
        startIntervalFromNow);
    if (null == timeSpanToStart)
        return "Invalid duration units." +
            "Valid options: seconds, minutes, hours, or days";

    string strPolicy = CreatePolicyStatement(
        pathToPolicyStmnt, urlString, DateTime.Now.Add(timeSpanToStart),
        DateTime.Now.Add(timeSpanInterval), ipaddress);

    // Read the policy into a byte buffer.
    byte[] bufferPolicy = Encoding.ASCII.GetBytes(strPolicy);

    // Convert the policy statement to URL-safe Base64 encoding and
    // replace unsafe characters + = / with the safe characters - _ ~

    string urlSafePolicy = ToUrlSafeBase64String(bufferPolicy);

    // Initialize the SHA1CryptoServiceProvider object and hash the policy data.

    byte[] bufferPolicyHash;
    using (SHA1CryptoServiceProvider cryptoSHA1 =
        new SHA1CryptoServiceProvider())
    {
        bufferPolicyHash = cryptoSHA1.ComputeHash(bufferPolicy);

        // Initialize the RSACryptoServiceProvider object.
        RSACryptoServiceProvider providerRSA = new RSACryptoServiceProvider();

        XmlDocument xmlPrivateKey = new XmlDocument();

        // Load PrivateKey.xml, which you created by converting your
        // .pem file to the XML format that the .NET framework uses.
        // Several tools are available. We used
        // .NET 2.0 OpenSSL Public and Private Key Parser,
        // http://www.jensign.com/opensslkey/opensslkey.cs.
```

```
xmlPrivateKey.Load("PrivateKey.xml");

// Format the RSACryptoServiceProvider providerRSA
// and create the signature.
providerRSA.FromXmlString(xmlPrivateKey.InnerXml);
RSAPKCS1SignatureFormatter RSAFormatter =
    new RSAPKCS1SignatureFormatter(providerRSA);
RSAFormatter.SetHashAlgorithm("SHA1");
byte[] signedHash = RSAFormatter.CreateSignature(bufferPolicyHash);

// Convert the signed policy to URL-safe Base64 encoding and
// replace unsafe characters + = / with the safe characters - _ ~
string strSignedPolicy = ToUrlSafeBase64String(signedHash);

return urlString +
    "?Policy=" +
    urlSafePolicy +
    "&Signature=" +
    strSignedPolicy +
    "&Key-Pair-Id=" +
    PrivateKeyId;
}
```

### Example Utility Methods for Signature Generation

The following methods get the policy statement from a file and parse time intervals for signature generation.

```
public static string CreatePolicyStatement(string policyStmnt,
    string resourceUrl,
    DateTime startTime,
    DateTime endTime,
    string ipAddress)
{
    // Create the policy statement.
    FileStream streamPolicy = new FileStream(policyStmnt, FileMode.Open,
    FileAccess.Read);
    using (StreamReader reader = new StreamReader(streamPolicy))
    {
        string strPolicy = reader.ReadToEnd();

        TimeSpan startTimeSpanFromNow = (startTime - DateTime.Now);
        TimeSpan endTimeSpanFromNow = (endTime - DateTime.Now);
        TimeSpan intervalStart =
            (DateTime.UtcNow.Add(startTimeSpanFromNow)) -
            new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);
        TimeSpan intervalEnd =
            (DateTime.UtcNow.Add(endTimeSpanFromNow)) -
            new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);

        int startTimestamp = (int)intervalStart.TotalSeconds; // START_TIME
        int endTimestamp = (int)intervalEnd.TotalSeconds; // END_TIME

        if (startTimestamp > endTimestamp)
            return "Error!";

        // Replace variables in the policy statement.
        strPolicy = strPolicy.Replace("RESOURCE", resourceUrl);
        strPolicy = strPolicy.Replace("START_TIME", startTimestamp.ToString());

        strPolicy = strPolicy.Replace("END_TIME", endTimestamp.ToString());
        strPolicy = strPolicy.Replace("IP_ADDRESS", ipAddress);
        strPolicy = strPolicy.Replace("EXPIRES", endTimestamp.ToString());
        return strPolicy;
    }
}

public static TimeSpan GetDuration(string units, string numUnits)
{
    TimeSpan timeSpanInterval = new TimeSpan();
    switch (units)
    {
        case "seconds":
            timeSpanInterval = new TimeSpan(0, 0, 0, int.Parse(numUnits));
            break;
        case "minutes":
            timeSpanInterval = new TimeSpan(0, 0, int.Parse(numUnits), 0);
            break;
        case "hours":
            timeSpanInterval = new TimeSpan(0, int.Parse(numUnits), 0, 0);
            break;
    }
}
```

```
        case "days":
            timeSpanInterval = new TimeSpan(int.Parse(numUnits), 0, 0, 0);
            break;
        default:
            Console.WriteLine("Invalid time units;" +
                "use seconds, minutes, hours, or days");
            break;
    }
    return timeSpanInterval;
}

private static TimeSpan GetDurationByUnits(string durationUnits,
    string startIntervalFromNow)
{
    switch (durationUnits)
    {
        case "seconds":
            return new TimeSpan(0, 0, int.Parse(startIntervalFromNow));
        case "minutes":
            return new TimeSpan(0, int.Parse(startIntervalFromNow), 0);
        case "hours":
            return new TimeSpan(int.Parse(startIntervalFromNow), 0, 0);
        case "days":
            return new TimeSpan(int.Parse(startIntervalFromNow), 0, 0, 0);
        default:
            return new TimeSpan(0, 0, 0, 0);
    }
}

public static string CopyExpirationTimeFromPolicy(string policyStatement)
{
    int startExpiration = policyStatement.IndexOf("EpochTime");
    string strExpirationRough = policyStatement.Substring(startExpiration +
        "EpochTime".Length);
    char[] digits = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9' };

    List<char> listDigits = new List<char>(digits);
    StringBuilder buildExpiration = new StringBuilder(20);

    foreach (char c in strExpirationRough)
    {
        if (listDigits.Contains(c))
            buildExpiration.Append(c);
    }
    return buildExpiration.ToString();
}
```

**See also**

- [Create a URL Signature Using Perl \(p. 110\)](#)
- [Create a URL Signature Using PHP \(p. 111\)](#)
- [Create a URL Signature Using Java \(p. 122\)](#)
- [GUI Tools for Signature Generation \(p. 125\)](#)

## Create a URL Signature Using Java

The [Open source Java toolkit for Amazon S3 and CloudFront](#) provides sample code and information about CloudFront development in Java. For information about private distributions, go to Private Distributions at [Programmer Guide: Code Samples](#).

### Note

Creating a URL signature is just one part of the process of serving private content using a signed URL. For more information about the entire process, see [How to Serve Private Content Using a Signed URL](#) (p. 88).

The following methods are from the Java open source toolkit for Amazon S3 and CloudFront. You must convert the private key from PEM to DER format for Java implementations to use it.

## Example Java Policy and Signature Encryption Methods

```
// Signed URLs for a private distribution
// Note that Java only supports SSL certificates in DER format, so you will
// need to
// convert your PEM-formatted file to DER format. To do this, you can use
// openssl:
// openssl pkcs8 -topk8 -nocrypt -in origin.pem -inform PEM -out new.der -outform
// DER
// So the encoder works correctly, you should also add the bouncy castle jar
// to your project and then add the provider.

Security.addProvider(new org.bouncycastle.jce.provider.BouncyCastleProvider());

String distributionDomain = "alb2c3d4e5f6g7.cloudfront.net";
String privateKeyFilePath = "/path/to/rsa-private-key.der";
String s3ObjectKey = "s3/object/key.txt";
String policyResourcePath = distributionDomain + "/" + s3ObjectKey;

// Convert your DER file into a byte array.

byte[] derPrivateKey = ServiceUtils.readInputStreamToBytes(new
    FileInputStream(privateKeyFilePath));

// Generate a "canned" signed URL to allow access to a
// specific distribution and object

String signedUrlCanned = CloudFrontService.signUrlCanned(
    "http://" + distributionDomain + "/" + s3ObjectKey, // Resource URL or Path

    keyPairId,      // Certificate identifier,
                    // an active trusted signer for the distribution
    derPrivateKey, // DER Private key data
    ServiceUtils.parseIso8601Date("2011-11-14T22:20:00.000Z") // DateLessThan
);
System.out.println(signedUrlCanned);

// Build a policy document to define custom restrictions for a signed URL.

String policy = CloudFrontService.buildPolicyForSignedUrl(
    policyResourcePath, // Resource path (optional, may include '*' and '?'
    wildcards)
    ServiceUtils.parseIso8601Date("2011-11-14T22:20:00.000Z"), // DateLessThan

    "0.0.0.0/0", // CIDR IP address restriction (optional, 0.0.0.0/0 means
    everyone)
    ServiceUtils.parseIso8601Date("2011-10-16T06:31:56.000Z") // DateGreaterThan
    (optional)
);

// Generate a signed URL using a custom policy document.

String signedUrl = CloudFrontService.signUrl(
    "http://" + distributionDomain + "/" + s3ObjectKey, // Resource URL or Path

    keyPairId,      // Certificate identifier, an active trusted signer for the
    distribution
    derPrivateKey, // DER Private key data
```

```
    policy // Access control policy  
    );  
System.out.println(signedUrl);
```

See also

- [Create a URL Signature Using Perl \(p. 110\)](#)
- [Create a URL Signature Using PHP \(p. 111\)](#)
- [Create a URL Signature Using C# and the .NET Framework \(p. 114\)](#)
- [GUI Tools for Signature Generation \(p. 125\)](#)



## GUI Tools for Signature Generation

The following third-party tools provide interfaces that accept user input and create CloudFront private distributions and URL signatures.

CloudBuddy	For information about using <a href="#">CloudBuddy</a> for CloudFront private content, see <a href="#">Configuring CloudFront Distribution and Private Content</a> . This tool is based on research at <a href="#">CSS CorpLabs</a> for a <a href="#">.NET implementation</a> of CloudFront private URLs.
Bucket Explorer	For information about using <a href="#">Bucket Explorer</a> for CloudFront private content, see <a href="#">How to Create a Private Distribution on a Bucket</a> .
CloudBerry	For information about using <a href="#">CloudBerry</a> for CloudFront private content, go to <a href="#">How to Configure Private Content for CloudFront Streaming with CloudBerry</a> .

See also

- [Create a URL Signature Using Perl](#) (p. 110)
- [Create a URL Signature Using PHP](#) (p. 111)
- [Create a URL Signature Using C# and the .NET Framework](#) (p. 114)
- [Create a URL Signature Using Java](#) (p. 122)

# Using an HTTPS Connection to Access Your Objects

---

## Topics

- [How CloudFront Works with HTTPS Connections \(p. 127\)](#)
- [How to Require HTTPS for Communication Between Viewers, CloudFront, and Your Origin \(p. 127\)](#)
- [Alternate Domain Names \(CNAMEs\) and HTTPS \(p. 128\)](#)
- [Charges for HTTPS Connections \(p. 128\)](#)

For download distributions, you can use HTTPS requests to ensure that your objects are encrypted when CloudFront serves them to viewers and, if you want, when CloudFront gets them from your origin. When you create a download distribution, you can:

- Configure one or more CloudFront cache behaviors to require that viewers use only the HTTPS protocol to access your objects in the CloudFront cache. This allows you to require HTTPS for some objects but not for others.
- Configure one or more CloudFront origins to require that CloudFront fetches objects from your origin using the protocol that the viewer used to request the objects. For example, when you use this CloudFront setting and when the viewer uses HTTPS to request an object from CloudFront, CloudFront also uses HTTPS to forward the request to your origin. When your origin is an Amazon S3 bucket, this is the default setting and cannot be changed.

If you're using an HTTP server as your origin, and if you want to use HTTPS both between viewers and CloudFront and between CloudFront and your origin, you must install an SSL certificate on the HTTP server that is signed by a third-party certificate authority, for example, VeriSign or DigiCert.

## Caution

If the origin server returns either an invalid certificate or a self-signed certificate, CloudFront drops the TCP connection.

You can, however, use a self-signed certificate if you only configure cache behaviors to require HTTPS between viewers and CloudFront, and you allow HTTP between CloudFront and the origin.

## How CloudFront Works with HTTPS Connections

The following example of how CloudFront works with HTTPS connections assumes the following:

- Your CloudFront distribution has one cache behavior (the default cache behavior) and one origin.
- You have configured your distribution to use HTTPS between viewers and CloudFront and between CloudFront and your origin.
- Your origin has an SSL certificate that was signed by a third-party certificate authority.

The process works basically the same way whether your origin server is an Amazon S3 bucket or an HTTP server.

### CloudFront Process for Serving Objects Using HTTPS

1. A viewer submits an HTTPS request to CloudFront. There's some SSL negotiation here between the viewer and CloudFront. In the end, the viewer submits the request in an encrypted format.
2. If the object is in the CloudFront edge cache, CloudFront encrypts the object and returns it to the viewer, and the viewer decrypts it.
3. If the object is not in the CloudFront cache, CloudFront performs the SSL negotiation with your origin and, when the negotiation is complete, forwards the request to your origin in an encrypted format.
4. Your origin decrypts the request, encrypts the requested object, and returns the object to CloudFront.
5. CloudFront decrypts the object, re-encrypts it, and forwards the object to the viewer. CloudFront also saves the object in the edge cache so the object is available next time it's requested.
6. The viewer decrypts the object.

## How to Require HTTPS for Communication Between Viewers, CloudFront, and Your Origin

You can configure CloudFront to require HTTPS for communication between viewers and CloudFront and, optionally, between CloudFront and your origin.

### Note

To ensure that objects are encrypted from the origin to CloudFront edge caches and from edge caches to viewers, use only HTTPS. If you ever configure CloudFront to get objects from your origin using HTTP, CloudFront adds the objects to the edge cache and continues to serve them to viewers until the objects expire, or until you remove or replace them. For more information about removing or replacing objects in a distribution, see [Adding, Removing, or Replacing Objects in a Distribution](#) (p. 64).

### To Require HTTPS for Communication Between Viewers, CloudFront, and Your Origin

1. If you're using an HTTP server as your origin and if you don't already have an SSL certificate for the server, get and install an SSL certificate from a third-party certificate authority such as VeriSign or DigiCert.

### Note

If you're using an Amazon S3 bucket, Amazon S3 provides an SSL certificate.

For more information about getting and installing an SSL certificate, refer to the documentation for your HTTP server software and to the documentation for the third-party certificate authority.

2. To require that viewers use HTTPS when communicating with CloudFront, create or update one or more cache behaviors in your distribution to have the following settings:

- **CloudFront Console:** For **Viewer Protocol Policy**, specify **HTTPS Only**.
- **CloudFront API:** For `ViewerProtocolPolicy`, specify `https-only`.

For information about using the CloudFront console to update a download distribution, see [Listing, Viewing, and Updating CloudFront Distributions \(p. 55\)](#).

For information about using the CloudFront API to update a download distribution, go to [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.

3. *Optional:* To require that CloudFront uses HTTPS when communicating with your origin, create or update one or more origins in your distribution to have the following settings:

- **CloudFront Console:** For **Origin Protocol Policy**, specify **Match Viewer**.
- **CloudFront API:** For `OriginProtocolPolicy`, specify `match-viewer`.

When your origin is an Amazon S3 bucket, **Match Viewer** is the default setting and cannot be changed.

4. Confirm the following:
  - The path pattern in each cache behavior applies only to the requests for which you want viewers to use HTTPS.
  - The cache behaviors are listed in the desired order. For more information, see [Path Pattern \(p. 42\)](#).
  - The cache behaviors are routing requests to the origins for which you have configured an **Origin Protocol Policy** of **Match Viewer**, if applicable.
  - The origins have valid certificates signed by a third-party certificate authority.

## Alternate Domain Names (CNAMEs) and HTTPS

If you want viewers to use HTTPS to access your objects, you must use the CloudFront domain name that is assigned to your distribution in your URLs. However, you can add one or more alternate domain names to a download distribution and then:

- Use HTTPS when the links to your objects use the CloudFront domain name, for example, `https://d1111111abcdef8.cloudfront.net/image.jpg`.
- Use alternate domain names when the links to your objects use HTTP, for example, `http://www.example.com/image/image.jpg`.

If you want to use both alternate domain names and HTTPS in the same CloudFront download distribution (but not for the same objects), you must create at least one cache behavior for which the value of **Viewer Protocol Policy** is **HTTP and HTTPS** (`allow-all` if you're using the CloudFront API).

## Charges for HTTPS Connections

You always incur a surcharge for HTTPS requests and bytes transferred. For information on billing rates, refer to the [CloudFront pricing plan](#).

# Access Logs

---

## Topics

- [Overview \(p. 129\)](#)
- [Bucket and File Ownership \(p. 131\)](#)
- [How to Change Logging Settings \(p. 131\)](#)
- [How to Delete Log Files from an Amazon S3 Bucket \(p. 132\)](#)
- [File Name Format and Timing of File Delivery \(p. 132\)](#)
- [Log File Format \(p. 132\)](#)
- [Charges for Access Logs \(p. 137\)](#)

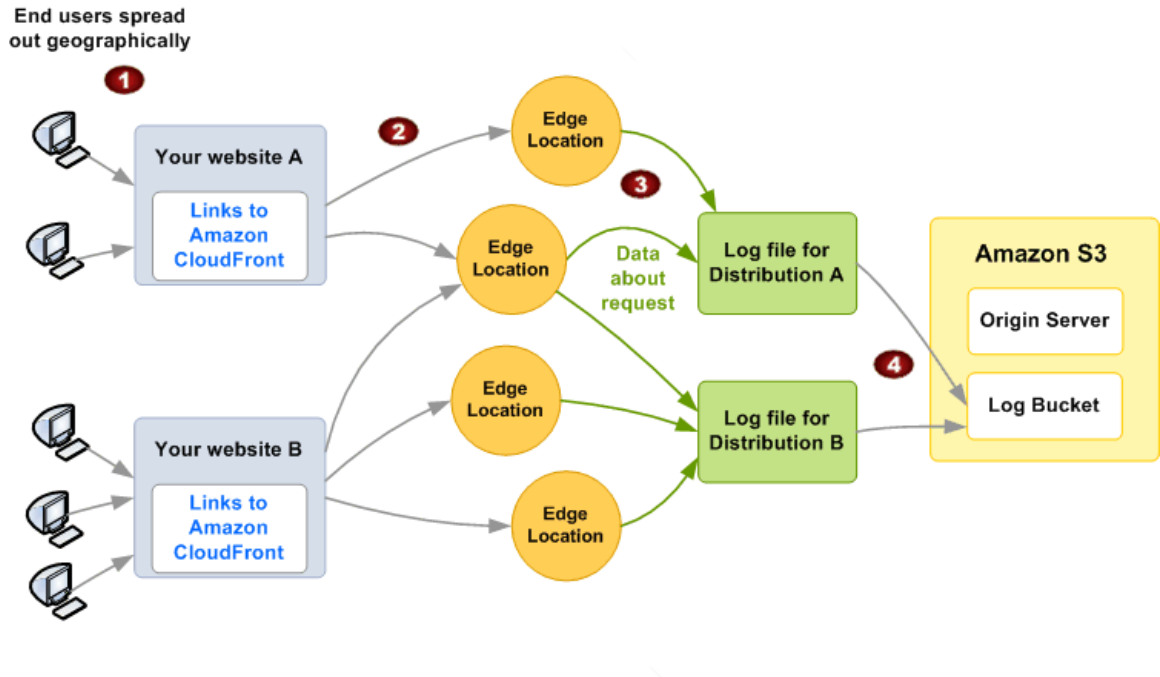
Amazon CloudFront provides optional log files with information about end user access to your objects. This section describes how to enable and disable logging, the content of log files, and how AWS charges you if you decide to use logging.

## Note

If you use a custom origin, you will need to create an Amazon S3 bucket to store your log files in.

## Overview

You can enable CloudFront to deliver access logs per distribution to an Amazon S3 bucket of your choice. The following figure and table describe the basic process for access logs.



### Process for Access Logs

<b>1</b>	Your end users use your application or website. In this graphic, you have two different websites, A and B, each using a different CloudFront distribution (Distribution A and Distribution B).
<b>2</b>	Your end users send requests for content, and CloudFront routes each request to the appropriate edge location.
<b>3</b>	CloudFront writes data about each request to a log file specific to that distribution. In this graphic, CloudFront writes information about requests related to Distribution A in a log file just for Distribution A, and requests for Distribution B in a log file just for Distribution B.
<b>4</b>	CloudFront periodically puts the distribution's log file in an Amazon S3 bucket of your choice, and then starts writing a new log file for the distribution.

Each entry in a log file gives details about a single end user request for an object. For more information about log file format, see [Log File Format \(p. 132\)](#).

You can store log files for a distribution either in the same Amazon S3 bucket as your origin server or in a different bucket. You can also store the log files for multiple distributions in the same bucket. When you enable logging for a particular distribution, you can specify an optional prefix for the names of your log files.

If a distribution has no end user requests during a particular hour, you don't receive a log file for that hour.

### Note

Because logs for a single stream can get recorded in multiple files, we recommend you combine all the log files you receive for a given period into one file. You can then analyze the data for that period more quickly and accurately.

### Important

You should use the logs to understand the nature of the requests for your content, not as a complete accounting of all requests. CloudFront delivers access logs on a best-effort basis. The log record for a particular request might be delivered long after the request was actually processed, or not at all. In rare cases, usage that appears in the AWS usage tracking and billing systems might not appear in CloudFront access logs.

## Bucket and File Ownership

You must have Amazon S3 `FULL_CONTROL` permission for the log file bucket. You have this permission by default if you're the bucket owner. If you're not, the bucket owner must grant your AWS account `FULL_CONTROL` permission.

When you enable logging, you do it with an API call to the CloudFront control API. Making that API call also automatically calls the Amazon S3 API to update the bucket's ACL to allow read and write permissions for the *AWS data feeds account*. This account writes the log files to the bucket.

Each log file has its own ACL (separate from the bucket's ACL). The bucket owner has `FULL_CONTROL` permission for the log files, the distribution owner (if not the bucket owner) has no permission, and the data feeds account has read and write permission.

### Note

Removing the permissions for the data feeds account does not disable logging. If you remove those permissions, but don't disable logging (which you do with the control API), we reinstate those permissions the next time the data feeds account needs to write a log file to your log bucket.

If you disable logging, we don't remove the read/write permissions for the data feeds account on either the bucket or the log files. If you want, you can do that yourself.

## How to Change Logging Settings

You can enable or disable logging, change the Amazon S3 bucket where your logs are stored, and change the prefix for log files using the CloudFront console or using the CloudFront API:

- For information about updating a download or a streaming distribution using the CloudFront console, see [Listing, Viewing, and Updating CloudFront Distributions](#) (p. 55).
- For information about updating a download distribution using the CloudFront API, go to [PUT Distribution Config](#) in the *Amazon CloudFront API Reference*.
- For information about updating a streaming distribution using the CloudFront API, go to [PUT Streaming Distribution Config](#) in the *Amazon CloudFront API Reference*.

Your changes to logging settings take effect within 12 hours.

To use the CloudFront API to change access log settings for:

- download distributions, you must use the 2009-04-02 or later version of the API.
- streaming distributions, you must use the 2010-05-01 or later version of the API.

### Note

To enable easier listing of keys in a bucket, Amazon S3 users commonly use a prefix followed by a slash (/) as a delimiter.

## How to Delete Log Files from an Amazon S3 Bucket

CloudFront does not automatically delete log files from the Amazon S3 bucket that you specified when you enabled logging. For information about deleting log files from an Amazon S3 bucket, see the applicable Amazon S3 documentation:

- Using the Amazon S3 console: See [Deleting an Object](#) in the *Amazon Simple Storage Service Console User Guide*.
- Using the REST API: See [DELETE Object](#) in the *Amazon Simple Storage Service API Reference*.
- Using the SOAP API: See [DeleteObject](#) in the *Amazon Simple Storage Service API Reference*.

## File Name Format and Timing of File Delivery

The filename follows this format (with the date and hour in UTC):

```
{Bucket}.s3.amazonaws.com/{Optional Prefix You Choose}{Distribution ID}.{YYYY}-{MM}-{DD}-{HH}.{Unique ID}.gz
```

For example, if your bucket name is `mylogs`, and you name your prefix `myprefix/`, your filenames look similar to this:

```
mylogs.s3.amazonaws.com/myprefix/EMLARXS9EXAMPLE.2009-03-17-20.RT4KCN4SGK9.gz
```

Log files are delivered to your bucket within 24 hours after an end user requests an object, and typically sooner. Log files generally arrive in your bucket once an hour.

Each hour of usage is typically covered in a single log file. CloudFront compresses the file in gzip format before delivering it to your bucket. CloudFront might write multiple files for a given hour of usage. For example, this occurs if the log file contents for the hour exceed 50 MB (uncompressed).

### Note

If a distribution has no end user requests during a particular hour, you don't receive a log file for that hour.

## Log File Format

Each entry in a log file gives details about a single end user request for an object. The log files for download and for streaming distributions are not identical, but they each:

- Use the W3C extended log file format. (For more information, go to <http://www.w3.org/TR/WD-logfile.html>.)
- Contain tab-separated values.
- Contain records that are not necessarily in chronological order.



- Contain two header lines: one with the file format version, and another that lists the W3C fields included in each record.
- Substitute URL-encoded equivalents for spaces and non-standard characters in field values.

These non-standard characters consist of all ASCII codes below 32 and above 127, plus the characters in the following table. The URL encoding standard is RFC 1738. For more information, go to <http://www.ietf.org/rfc/rfc1738.txt>.

Hexadecimal Value	Character
0x3C	<
0x3E	>
0x22	"
0x23	#
0x25	%
0x7B	{
0x7D	}
0x7C	
0x5C	\
0x5E	^
0x7E	~
0x5B	[
0x5D	]
0x60	`
0x27	'
0x20	space

## Download Distribution File Format

The following table describes the fields for one record in the download distribution log file.

Field	Description
c-ip	Client IP, for example, 192.0.2.183.
cs(Host)	DNS name (the CloudFront distribution name specified in the request). If you made the request to a CNAME, the DNS name field will contain the underlying distribution DNS name, not the CNAME.

Field	Description
cs-method	HTTP access method.
cs(Referer)	The referrer.
cs(User-Agent)	The user agent.
date	The date (UTC) on which the event occurred, for example, 2009-03-10.
s-uri-stem	URI stem (e.g., /images/daily-ad.jpg).
sc-bytes	Server to client bytes, for example, 1045619.
sc-status	HTTP status code (e.g., 200).
time	Time when the server finished processing the request (UTC), for example, 01:42:39.
x-edge-location	The edge location that served the request. Each edge location is identified by a three-letter code and an arbitrarily assigned number, for example, DFW3. The three-letter code typically corresponds with the International Air Transport Association airport code for an airport near the edge location. (These abbreviations may change in the future.) For a list of edge locations, see the Amazon CloudFront detail page, <a href="http://aws.amazon.com/cloudfront">http://aws.amazon.com/cloudfront</a> .
cs-uri-query	The query string portion of the URI that is included on the connect string. When a URI doesn't contain a query string, the log file contains a single dash (-). The log records query strings to a maximum length of 8K bytes. The encoding standard is RFC 1738, as described in <a href="#">Log File Format (p. 132)</a>

### Note

Question marks (?) in URLs and query strings are not included in the log.

The fields appear in the following order in a record:

- date
- time
- x-edge-location
- sc-bytes
- c-ip
- cs-method
- cs(Host)
- cs-uri-stem
- sc-status
- cs(Referer)
- cs(User Agent)
- cs-uri-query

The following is an example log file for a download distribution.

```
#Version: 1.0
#Fields: date time x-edge-location sc-bytes c-ip cs-method cs(Host) cs-uri-stem
        sc-status cs(Referer) cs(User-Agent) cs-uri-query
02/01/2011 01:13:11 FRA2 182 192.0.2.10 GET d2819bc28.cloudfront.net
```

```
/view/my/file.html 200 www.displaymyfiles.com Mozilla/4.0%20(compatible;%20MSIE%205.0b1;%20Mac_PowerPC) -  
02/01/2011 01:13:12 LAX1 2390282 192.0.2.202 GET www.singalong.com  
/soundtrack/happy.mp3 304 www.unknownsingers.com Mozilla/4.0%20(compatible;%20MSIE%207.0;%20Windows%20NT%205.1) a=b&c=d
```

## Streaming Distribution Log File Format

Each record in a streaming access log represents a playback event, for example, connect, play, pause, stop, disconnect, and so on. So, CloudFront generates multiple log records each time a viewer watches a video. To relate log records that stem from the same stream ID, use the *x-sid* field.

### Note

Some fields are present for all events, whereas others appear only on Play, Stop, Pause, Unpause, and Seek events. When a field isn't relevant for a given event, the log file will contain a single dash (-).

The following table describes the fields that are present on each record in the streaming distribution log file, regardless of the type of event.

Field	Description
date	Date (UTC) on which the event occurred.
time	Time when the server received the request (UTC), for example, 01:42:39.
x-edge-location	The edge location where the playback event occurred. Each edge location is identified by a three-letter code and an arbitrarily assigned number, for example, DFW3. The three-letter code typically corresponds with the International Air Transport Association airport code for an airport near the edge location. (These abbreviations may change in the future.) For a list of edge locations, see the Amazon CloudFront detail page, <a href="http://aws.amazon.com/cloudfront">http://aws.amazon.com/cloudfront</a> .
c-ip	Client IP, for example, 192.0.2.183.
x-event	The event type. This is a Connect, Disconnect, Play, Stop, Pause, Unpause, or Seek event.
sc-bytes	The running total number of bytes sent from the server to the client, up to the time of the event.
x-cf-status	A code indicating the status of the event. Currently, "OK" is the only value for this field. New functionality in the future could require new status codes.
x-cf-client-id	An opaque string identifier that can be used to differentiate clients. This value is unique for each connection.
cs-uri-stem	The stem portion of the URI, including the application and the application instance. This is sometimes referred to as the FMS <i>connect string</i> . For example, <code>rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st</code> .
cs-uri-query	The query string portion of the URI that is included on the connect string.
c-referrer	The URI of the referrer.
x-page-url	The URL of the page from which the SWF is linked.

Field	Description
c-user-agent	The user agent.

The following fields are present only on Play, Stop, Pause, Unpause, and Seek events. For other events, these fields will contain a single dash (-).

Field	Description
x-sname	The stream name.
x-sname-query	The stream query string, if any.
x-file-ext	The stream type, for instance, FLV.
x-sid	The stream ID. This is a unique integer identifier for the connection.

### Note

Question marks (?) in URLs and query strings are not included in the log.

The fields appear in the following order in a record:

- date
- time
- x-edge-location
- c-ip
- x-event
- sc-bytes
- x-cf-status
- x-cf-client-id
- cs-uri-stem
- cs-uri-query
- c-referrer
- x-page-url
- c-user-agent
- x-sname
- x-sname-query
- x-file-ext
- x-sid

The following is an example of a log file for a streaming distribution.

```
#Version: 1.0
#Fields: date time x-edge-location c-ip x-event sc-bytes x-cf-status x-cf-client-id cs-uri-stem cs-uri-query c-referrer x-page-url c-user-agent x-sname x-sname-query x-file-ext x-sid
2010-03-12 23:51:20 SEA4 192.0.2.147 connect 2014 OK
bfd8a98bee0840d9b871b7f6ade9908f rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st
key=value http://player.longtailvideo.com/player.swf http://www.long
tailvideo.com/support/jw-player-setup-wizard?example=204 LNX%2010,0,32,18
```

```
- - - -
2010-03-12 23:51:21 SEA4 192.0.2.222 play 3914 OK
bfd8a98bee0840d9b871b7f6ade9908f rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st
key=value http://player.longtailvideo.com/player.swf http://www.long
tailvideo.com/support/jw-player-setup-wizard?example=204 LNX%2010,0,32,18
myvideo p=2&q=4 flv 1
2010-03-12 23:53:44 SEA4 192.0.2.4 stop 323914 OK
bfd8a98bee0840d9b871b7f6ade9908f rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st
key=value http://player.longtailvideo.com/player.swf http://www.long
tailvideo.com/support/jw-player-setup-wizard?example=204 LNX%2010,0,32,18
dir/other/myvideo p=2&q=4 flv 1
2010-03-12 23:53:44 SEA4 192.0.2.103 play 8783724 OK
bfd8a98bee0840d9b871b7f6ade9908f rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st
key=value http://player.longtailvideo.com/player.swf http://www.long
tailvideo.com/support/jw-player-setup-wizard?example=204 LNX%2010,0,32,18
dir/favs/myothervideo p=42&q=14 mp4 2
2010-03-12 23:56:21 SEA4 192.0.2.199 stop 429822014 OK
bfd8a98bee0840d9b871b7f6ade9908f rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st
key=value http://player.longtailvideo.com/player.swf http://www.long
tailvideo.com/support/jw-player-setup-wizard?example=204 LNX%2010,0,32,18
dir/favs/myothervideo p=42&q=14 mp4 2
2010-03-12 23:59:44 SEA4 192.0.2.14 disconnect 429824092 OK
bfd8a98bee0840d9b871b7f6ade9908f rtmp://shqshne4jdp4b6.cloudfront.net/cfx/st
key=value http://player.longtailvideo.com/player.swf http://www.long
tailvideo.com/support/jw-player-setup-wizard?example=204 LNX%2010,0,32,18
- - - -
```

## Charges for Access Logs

Access logging is an optional feature of CloudFront. There is no extra charge for enabling access logging. However, you accrue the usual Amazon S3 charges for storing and accessing the files on Amazon S3 (you can delete them at any time).

### Related Topics

- [Paying for CloudFront \(p. 8\)](#)

# General Usage Data

---

In addition to the optional CloudFront access logs (for more information, see [Access Logs \(p. 129\)](#)), you can get other, more general information about your AWS service usage at no cost.

## AWS Account Activity

You can get general information about your AWS service usage and costs on your Account Activity page.

### To access your account activity

- Go to <http://aws.amazon.com>, click **Your Account**, and then select **Account Activity**.

## Usage Report

AWS provides a usage report for Amazon CloudFront, similar to the one for Amazon S3. You can download aggregate usage by edge location region (Europe, U.S., Japan, Asia-Pacific) and by usage type (data transferred out and requests). You can aggregate the data by hour, day, or month. If you want information about the GETS from your origin server, refer to the usage report for Amazon S3.

### To get usage reports

- Go to <http://aws.amazon.com>, click **Your Account**, and then select **Usage Reports**.

### Note

The value for the `Resource` field in the CloudFront usage report is your distribution's ID.

# Troubleshooting

---

If you cannot view your CloudFront distribution, the following topics describe some common solutions.

## Topics

- [Did you sign up for both CloudFront and Amazon S3? \(p. 139\)](#)
- [Are your Amazon S3 bucket and object permissions set correctly? \(p. 139\)](#)
- [Is your alternate domain name \(CNAME\) correctly configured? \(p. 140\)](#)
- [Are you referencing the correct URL for your CloudFront distribution? \(p. 140\)](#)
- [If you are working with a streaming distribution, are your URL and your playback client correctly configured? \(p. 141\)](#)
- [Do you need help troubleshooting a custom origin? \(p. 141\)](#)

## Did you sign up for both CloudFront and Amazon S3?

To use Amazon CloudFront with an Amazon S3 origin, you must sign up for both CloudFront and Amazon S3, separately. For more information about signing up for CloudFront and Amazon S3, see [Getting Started with CloudFront \(p. 14\)](#).

## Are your Amazon S3 bucket and object permissions set correctly?

If you are using CloudFront with an Amazon S3 origin, the original versions of your content are stored in an Amazon S3 bucket. The easiest way to use CloudFront with Amazon S3 is to make all your objects publicly readable in Amazon S3. To do this, you must explicitly enable public read privileges for each object you upload to Amazon S3.

If your content is not publicly readable, you need to create a CloudFront origin access identity so CloudFront can access it. For more information about CloudFront origin access identities, see [Securing Your Content in Amazon S3 \(p. 90\)](#).

Object properties and bucket properties are independent. You must explicitly grant privileges to each object in Amazon S3. Objects do not inherit properties from buckets and object properties must be set independently of the bucket.

## Is your alternate domain name (CNAME) correctly configured?

If you already have an existing CNAME record for your domain name, update that record or replace it with a new one that points to your distribution's domain name.

Also, make sure your CNAME record points to your distribution's domain name, not your Amazon S3 bucket. You can confirm that the CNAME record in your DNS system points to your distribution's domain name. To do so, use a DNS tool like dig. (For information about dig, go to <http://www.kloth.net/services/dig.php>.)

The following shows an example dig request on a domain name called `images.example.com`, and the relevant part of the response. Under `ANSWER SECTION`, see the line that contains `CNAME`. The CNAME record for your domain name is set up correctly if the value on the right side of `CNAME` is your CloudFront distribution's domain name. If it's your Amazon S3 origin server bucket or some other domain name, then the CNAME record is set up incorrectly.

```
[prompt]> dig images.example.com

; <<> DiG 9.3.3rc2 <<> images.example.com
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 15917
;; flags: qr rd ra; QUERY: 1, ANSWER: 9, AUTHORITY: 2, ADDITIONAL: 0
;; QUESTION SECTION:
;images.example.com.      IN      A
;; ANSWER SECTION:
images.example.com. 10800 IN  CNAME  d111111abcdef8.cloudfront.net.
...
...
```

For more information about CNAMEs, see [Using Alternate Domain Names \(CNAMEs\)](#) (p. 53).

## Are you referencing the correct URL for your CloudFront distribution?

Make sure the URL you're referencing uses your CloudFront distribution domain name (or your CNAME), not your Amazon S3 bucket or custom origin.



## If you are working with a streaming distribution, are your URL and your playback client correctly configured?

Streaming distributions require you to use an RTMP protocol instead of HTTP, and you must make a few minor configuration changes to your playback client. For information about creating streaming distributions, see [Creating Streaming Distributions \(p. 28\)](#).

## Do you need help troubleshooting a custom origin?

If you need AWS to help you troubleshoot a custom origin, we will probably need to inspect the `X-Amz-Cf-Id` header entries from your requests. If you are not already logging these entries, you might want to consider it for the future. For more information, see [Requirements and Recommendations for Using Amazon EC2 and Other Custom Origins \(p. 45\)](#).

# Making API Requests

---

## Topics

- [Endpoints](#) (p. 142)
- [AWS Support for Programming Languages](#) (p. 143)
- [REST Requests](#) (p. 143)
- [REST Responses](#) (p. 146)
- [Authenticating REST Requests](#) (p. 148)

This section describes how to make REST requests to the CloudFront API, which you use to create and manage your distributions. The various topics acquaint you with the components of requests, the content of responses, and how to authenticate requests.

## Endpoints

Unlike other Amazon services, CloudFront doesn't have multiple endpoints based on the regions in which the service operates (e.g., Singapore, EU/Dublin, US/East, and so on). This is because CloudFront distributions aren't regional resources like Amazon S3 buckets and Amazon EC2 instances. Instead, Amazon CloudFront has the ability to serve content from one of its many edge locations. This means that CloudFront distributions have a single endpoint: the location of the origin server for a specific distribution.

As a result, when you make a REST request you use the following format, where *<distribution>* is the distribution that you are asking to take action on in your request.

```
cloudfront.amazonaws.com/2012-05-05/<distribution>
```

## Related Topics

- [REST Requests](#) (p. 143)
- [The Amazon CloudFront Network](#) (a list on the AWS website of all the Amazon CloudFront edge locations)
- [Regions and Endpoints](#) (information about AWS product endpoints and regions in the Amazon Web Services General Reference)

# AWS Support for Programming Languages

AWS provides libraries, sample code, tutorials, and other resources for software developers who prefer to build applications using language-specific APIs instead of CloudFront's REST API. These libraries provide basic functions (not included in CloudFront's REST API), such as request authentication, request retries, and error handling so you can get started more easily. Libraries and resources are available for the following languages:

- [Java](#)
- [PHP](#)
- [Ruby](#)
- [Windows and .NET](#)

For libraries and sample code in all languages, go to [Sample Code & Libraries](#).

## REST Requests

Amazon CloudFront REST requests are HTTPS requests, as defined by RFC 2616 (for more information, go to <http://www.ietf.org/rfc/rfc2616.txt>). This section describes the structure of a CloudFront REST request. For detailed descriptions of the actions you can perform, go to the [Amazon CloudFront API Reference](#).

A typical REST action consists of sending a single HTTPS request to CloudFront, and waiting for the HTTP response. Like any HTTP request, a REST request to CloudFront contains a request method, a URI, request headers, and sometimes a query string or request body. The response contains an HTTP status code, response headers, and sometimes a response body.

## Request URI

The request URI always starts with a forward slash and then the version of the CloudFront API you use (for example, 2012-05-05). The remainder of the URI indicates the particular resource you want to act on. For example, following is the URI you use when creating a new distribution (for more information, go to [POST Distribution](#) in the *Amazon CloudFront API Reference*).

`/2012-05-05/distribution`

The topics in this guide that describe the different API actions show how to structure the URI.

## Request Headers

The following table lists the HTTP headers that CloudFront REST requests use.

Header Name	Description	Required
Authorization	The information required for request authentication ().	Yes
Content-Length	Length of the message (without the headers) according to RFC 2616. Condition: Required if the request body itself contains information (most toolkits add this header automatically).	Conditional

Header Name	Description	Required
Content-Type	The content type of the resource. Example: <code>text/plain</code> . Condition: Required for POST and PUT requests.	Conditional
Date	The date used to create the signature contained in the <code>Authorization</code> header. The format must be one of the full date formats specified in RFC 2616 section 3.1.1. For example: <code>Wed, 05 Apr 2006 21:12:00 GMT</code> . For more information, go to <a href="#">the RFC 2616 specification</a> . Condition: Required unless you provide the <code>x-amz-date</code> header (for more information about the request time stamp, ).	Conditional
Host	The host being requested. The value must be <code>cloudfront.amazonaws.com</code> Condition: Required for HTTP 1.1 (most toolkits add this header automatically)	Conditional
x-amz-date	The date used to create the signature contained in the <code>Authorization</code> header. The format must be one of the full date formats specified in RFC 2616 section 3.1.1. For example: <code>Wed, 05 Apr 2006 21:12:00 GMT</code> . For more information, go to the <a href="#">RFC 2616 specification</a> . Condition: Required if you do not provide the <code>Date</code> header (for more information, ).	Conditional

## Request Time Stamp

You must provide the time stamp in either the HTTP `Date` header or the AWS `x-amz-date` header (some HTTP client libraries don't let you set the `Date` header). When an `x-amz-date` header is present, the system ignores any `Date` header when authenticating the request.

The time stamp must be within 15 minutes of the AWS system time when the request is received. If it isn't, the request fails with the `RequestExpired` error code. This is to prevent replays of your requests by an adversary.

## Request Body

Many of the CloudFront API actions require you to include XML in the body of the request. The XML conforms to the CloudFront schema. The topics in this guide that describe the API actions show the structure of the XML required in the request.

## Example Request

The following example request creates a distribution in the CloudFront system.

```
POST /2012-05-05/distribution HTTP/1.1
Host: cloudfront.amazonaws.com
Authorization: [AWS authentication string]
Date: Thu, 17 May 2012 19:37:58 GMT
[Other required headers]
```

```
<?xml version="1.0" encoding="UTF-8"?>
<DistributionConfig xmlns="http://cloudfront.amazonaws.com/doc/2012-05-05/">
  <CallerReference>example.com2012-04-11-5:09pm</CallerReference>
  <Aliases>
    ...
  </Aliases>
  <DefaultRootObject>index.html</DefaultRootObject>
  <Origins>
    ...
  </Origins>
  <CacheBehaviors>
    ...
  </CacheBehaviors>
  <Comment>example comment</Comment>
  <Logging>
    ...
  </Logging>
  <Enabled>true</Enabled>
</DistributionConfig>
```

### Related Topics

- [Authenticating REST Requests \(p. 148\)](#)
- [REST Responses \(p. 146\)](#)

# REST Responses

Amazon CloudFront responses are just standard HTTP responses. Some of the CloudFront actions return special information specific to CloudFront in the form of an HTTP header or XML in the body of the response. The specific details are covered in the API reference topic for the particular action.

## Request ID

Each response contains a request ID that you can use if you need to troubleshoot a request with AWS. The ID is contained in an HTTP header called `x-amz-request-id`. An example of a request ID is `647cd254-e0d1-44a9-af61-1d6d86ea6b77`.

## Example Response

The following example shows the response when creating a distribution.

```
201 Created
Location: https://cloudfront.amazonaws.com/2012-05-05/distribution/EDFDVBD6EXAMPLE
x-amz-request-id: request_id

<?xml version="1.0" encoding="UTF-8"?>
<Distribution xmlns="http://cloudfront.amazonaws.com/doc/2012-05-05/">
  <Id>EDFDVBD6EXAMPLE</Id>
  <Status>InProgress</Status>
  <LastModifiedTime>2012-05-19T19:37:58Z</LastModifiedTime>
  <DomainName>d111111abcdef8.cloudfront.net</DomainName>
  <DistributionConfig>
    <CallerReference>example.com2012-04-11-5:09pm</CallerReference>
    <Aliases>
      ...
    </Aliases>
    <DefaultRootObject>index.html</DefaultRootObject>
    <Origins>
      ...
    </Origins>
    <CacheBehaviors>
      ...
    </CacheBehaviors>
    <Comment>example comment</Comment>
    <Logging>
      ...
    </Logging>
    <Enabled>true</Enabled>
  </DistributionConfig>
</Distribution>
```

## Error Responses

If a REST request results in an error, the HTTP reply has:

- An XML error document as the response body
- Content-Type header: `application/xml`

- An appropriate 3xx, 4xx, or 5xx HTTP status code

Following is an example of the XML error document in a REST error response.

```
<ErrorResponse xmlns="http://cloudfront.amazonaws.com/doc/2012-05-05/">
  <Error>
    <Type>Sender</Type>
    <Code>InvalidURI</Code>
    <Message>Could not parse the specified URI.</Message>
  </Error>
  <RequestId>410c2a4b-e435-49c9-8382-3770d80d7d4c</RequestId>
</ErrorResponse>
```

### Related Topics

- [Errors](#) (in the *Amazon CloudFront API Reference*)
- [REST Requests](#) (p. 143)
- [Authenticating REST Requests](#) (p. 148)

# Authenticating REST Requests

## Topics

- [Comparison with Amazon S3](#) (p. 148)
- [Overview of the Authentication Process](#) (p. 148)
- [The String to Sign](#) (p. 149)
- [Calculating the Signature](#) (p. 149)
- [The Authorization Header](#) (p. 150)
- [Authentication Errors](#) (p. 150)
- [Fetching the Date](#) (p. 151)

Every request you make to the Amazon CloudFront control API must be authenticated. AWS and others in the coding community provide tools that automatically sign your requests as required for CloudFront. For more information, see [Where Do I Go from Here?](#) (p. 218) and the [CloudFront sample code and libraries](#) page. If you plan to write your own code to sign requests, then read this topic.

## Comparison with Amazon S3

If you already know how authentication works for Amazon S3 REST requests, then the information in this topic will be familiar to you. Here are the main differences between how you authenticate CloudFront and Amazon S3 requests:

- For CloudFront, you must use HTTPS
- For CloudFront, the canonical string to sign is simply the value of the `Date` header (or the `x-amz-date` header if you include it in the request)

Therefore, the value of the `Authorization` header is as follows:

```
Authorization: "AWS" + " " + AWSAccessKeyID + ":" +  
              Base64(HMAC-SHA1(UTF-8(Date), UTF-8(AWSSecretAccessKey)))
```

## Overview of the Authentication Process

Authentication is how you prove your identity to the system. You must prove your identity in all your requests to the CloudFront control API. The following sections describe how.

The CloudFront REST API uses a custom HTTP scheme based on a keyed-HMAC (Hash Message Authentication Code) for authentication. The following table describes the basic process for authentication.

### Process for Request Authentication

1	You create a string based on specific information in the request. For more information, see <a href="#">The String to Sign</a> (p. 149).
2	<p>You calculate a <i>signature</i> using your AWS Secret Access Key, the string from task 1, and an HMAC-SHA1 algorithm.</p> <p>Informally, we call this process <i>signing the request</i>, and we call the output of the HMAC algorithm the signature because it simulates the security properties of a real signature. For instructions on creating the signature, see <a href="#">Calculating the Signature</a> (p. 149).</p>



3	You include the signature in the request and send the request to AWS using HTTPS (HTTP requests are not accepted). For information about where to put the signature in the request, see <a href="#">The Authorization Header (p. 150)</a> .
4	We check your signature. When we receive your request, we fetch the AWS Secret Access Key that you claim to have and use it in the same way you did to compute a signature for the message. We then compare the signature we calculated to the signature you presented in the request. If the two signatures match, we accept and process the request. Otherwise, we reject the request and respond with an error message (for more information, see <a href="#">Authentication Errors (p. 150)</a> ).

#### Note

We also confirm the request time stamp is within 15 minutes of the AWS server time. For more information, see [Fetching the Date \(p. 151\)](#).

## The String to Sign

In the first task in the preceding process, you form a string. The string is simply the UTF-8 encoded value of the `Date` header in the request (e.g., `Thu, 17 May 2012 19:37:58 GMT`). Your request must include either the `Date` header, the `x-amz-date` header, or both (if both are present, we ignore the `Date` header when authenticating the request). You might decide to include the `x-amz-date` header if your HTTP client doesn't let you set the `Date` header.

The format you use for the header value must be one of the full date formats specified in RFC 2616, section 3.3.1, for example, `Wed, 05 Apr 2006 21:12:00 GMT`. For more information, go to the [RFC 2616 specification](#).

## Calculating the Signature

Calculating the value to include in the request is a simple procedure.

#### Calculating the Signature

1. Calculate an RFC 2104-compliant HMAC-SHA1 hash, using the string (see [The String to Sign \(p. 149\)](#)) and your Secret Access Key as the key.
2. Convert the resulting value to base64.  
The result is the signature you include in the request.

The following table shows a string, a fake Secret Access Key, and an example of a Base64-encoded signature. Note that the credentials have been obfuscated, so calculating using these values does not produce the Base64-encoded signature listed in the table.

String	Thu, 17 May 2012 17:08:48 GMT
Secret Access Key	wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Base64-encoded signature	4cP3hCesdCQJ1jP11111YSu0g=EXAMPLE

## The Authorization Header

To pass the signature to AWS, you include it as part of the standard HTTP `Authorization` header. You include both the signature and your AWS Access Key ID in the header using the following format:

```
Authorization: AWS <AWSAccessKeyId>:<Signature>
```

Note that there is a space after AWS.

Following is an example REST request with the example signature calculated in the preceding section. The AWS Access Key ID (AKIAIOSFODNN7EXAMPLE) is fake.

```
POST /2012-05-05/distribution HTTP/1.1
Host: cloudfront.amazonaws.com
Date: Thu, 14 Aug 2008 17:08:48 GMT
Authorization: AWS AKIAIOSFODNN7EXAMPLE:11111112222222jPXo7+e/YSu0g=
[Other required headers]

<?xml version="1.0" encoding="UTF-8"?>
<DistributionConfig xmlns="http://cloudfront.amazonaws.com/doc/2012-05-05/">
  <CallerReference>example.com2012-04-11-5:09pm</CallerReference>
  <Aliases>
    ...
  </Aliases>
  <DefaultRootObject>index.html</DefaultRootObject>
  <Origins>
    ...
  </Origins>
  <CacheBehaviors>
    ...
  </CacheBehaviors>
  <Comment>example comment</Comment>
  <Logging>
    ...
  </Logging>
  <Enabled>true</Enabled>
</DistributionConfig>
```

## Authentication Errors

If the signature we create based on your request and Secret Access Key doesn't match the signature you sent in the request, we return the following error.

```
<ErrorResponse xmlns="http://cloudfront.amazonaws.com/doc/2012-05-05/">
  <Error>
    <Type>Sender</Type>
    <Code>SignatureDoesNotMatch</Code>
    <Message>The request signature we calculated
      does not match the signature you provided.
      Check your AWS Secret Access Key and signing
      method. Consult the service documentation for details.
    </Message>
  </Error>
```

```
<RequestId>all170c87-d04d-47c9-964f-54e1a4883f4e</RequestId>  
</ErrorResponse>
```

## Fetching the Date

To avoid replays of your requests, AWS requires the time stamp in the request to be within 15 minutes of the AWS system time. To avoid clock synchronization errors, we recommend you fetch the current date from the CloudFront server and then use that as the time stamp for your request and the string for your signature.

### To fetch the date

- Send an unauthenticated GET request for the date resource.

```
GET /date HTTP/1.1  
Host: cloudfront.amazonaws.com
```

We return the current server date as the value of the `Date` response header (note that the HTTP status code may or may not be a 200). The date uses the RFC 1123 format (e.g., Wed, 18 Nov 2009 17:08:48 GMT). For more information, go to [the RFC 1123 specification](#).

# CloudFront Tutorials

---

The following tutorials explain how to use CloudFront for live streaming and for geoblocking:

- [Live HTTP Streaming Using CloudFront and Adobe Flash Media Server 4.5 \(p. 152\)](#)
- [Live Smooth Streaming Using Amazon CloudFront and IIS Media Services 4.1 \(p. 174\)](#)
- [Restricting Access to Files in a CloudFront Distribution Based on Geographic Location \(Geoblocking\) \(p. 191\)](#)

## Live HTTP Streaming Using CloudFront and Adobe Flash Media Server 4.5

### Topics

- [Overview of Live HTTP Streaming with Amazon Web Services \(p. 152\)](#)
- [Creating an Amazon Web Services Account \(p. 153\)](#)
- [Creating an Amazon EC2 Key Pair \(p. 154\)](#)
- [Subscribing to Adobe Flash Media Server \(p. 155\)](#)
- [Creating an AWS CloudFormation Stack for Live Streaming \(p. 155\)](#)
- [Verifying that Adobe Flash Media Server Is Running \(p. 160\)](#)
- [Setting Up Adobe Flash Media Live Encoder to Publish a Live Stream \(p. 161\)](#)
- [Embedding Flash Media Playback for an Amazon CloudFront Live HTTP Stream in a Web Application \(p. 164\)](#)
- [Deleting an AWS CloudFormation Stack for Live Streaming \(p. 166\)](#)
- [Frequently Asked Questions \(p. 167\)](#)
- [Additional Documentation \(p. 172\)](#)

## Overview of Live HTTP Streaming with Amazon Web Services

Live streaming with Amazon Web Services allows you to use the features of Adobe Flash Media Server version 4.5, including live video streaming where your live video is delivered by a series of HTTP requests

from the player that is controlled by manifest files. Flash Media Server 4.5 supports two HTTP file formats: HLS (HTTP Live Streaming) for iOS devices and HDS (HTTP Dynamic Streaming) for Flash applications. You can stream high-quality media using the free Flash Media Live Encoder desktop application either for Windows or for Mac OS.

Video is broken up into a series of smaller files (called segments or fragments) that are cached in the CloudFront network for improved performance. The live streams are then delivered to both Flash Player-compatible and iOS devices through standard HTTP caching and network infrastructures using the integrated HTTP server as an origin service in Flash Media Server 4.5. To review the new capabilities of Adobe Flash Media Server 4.5, go to [What's New in Flash Media Server 4.5](#) on the Adobe website.

To set up live streaming with Amazon Web Services (AWS), review the system requirements for Adobe Flash Player at <http://www.adobe.com/products/flashplayer/systemreqs/>. Then perform the procedures in the following sections:

1. [Creating an Amazon Web Services Account \(p. 153\)](#)
2. [Creating an Amazon EC2 Key Pair \(p. 154\)](#)
3. [Subscribing to Adobe Flash Media Server \(p. 155\)](#)
4. [Creating an AWS CloudFormation Stack for Live Streaming \(p. 155\)](#)
5. [Verifying that Adobe Flash Media Server Is Running \(p. 160\)](#)
6. [Setting Up Adobe Flash Media Live Encoder to Publish a Live Stream \(p. 161\)](#)
7. [Embedding Flash Media Playback for an Amazon CloudFront Live HTTP Stream in a Web Application \(p. 164\)](#)
8. [Deleting an AWS CloudFormation Stack for Live Streaming \(p. 166\)](#)

For frequently asked questions, see [Frequently Asked Questions \(p. 167\)](#).

For links to additional Adobe and AWS documentation, see [Additional Documentation \(p. 172\)](#).

For information about the Adobe Flash Media Server features available on Amazon Web Services, see <http://www.adobe.com/products/flashmediaserver/amazonwebservices>.

## Creating an Amazon Web Services Account

If you already have an AWS account, skip to [Creating an Amazon EC2 Key Pair \(p. 154\)](#). If you don't already have an AWS account, use the following procedure to create one.

### Note

When you create an account, AWS automatically signs up the account for all services. You are charged only for the services you use.

### To create an AWS account

1. Go to <http://aws.amazon.com>, and click **Create an AWS Account**.
2. Follow the on-screen instructions.

Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

Next: [Creating an Amazon EC2 Key Pair \(p. 154\)](#)

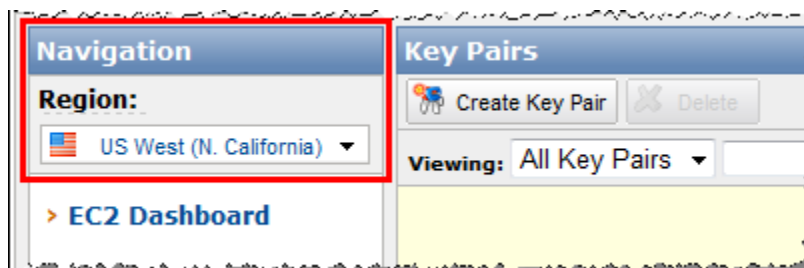
## Creating an Amazon EC2 Key Pair

If you already have an Amazon EC2 key pair in the Amazon EC2 region in which you want to configure live streaming, skip to [Subscribing to Adobe Flash Media Server \(p. 155\)](#). If you don't have a key pair in that region, perform the following procedure.

A key pair is a security credential similar to a password. You specify a key pair when you create an AWS CloudFormation stack for live streaming, later in this process. After live streaming is configured, you use the key pair to securely connect to your Amazon EC2 instance.

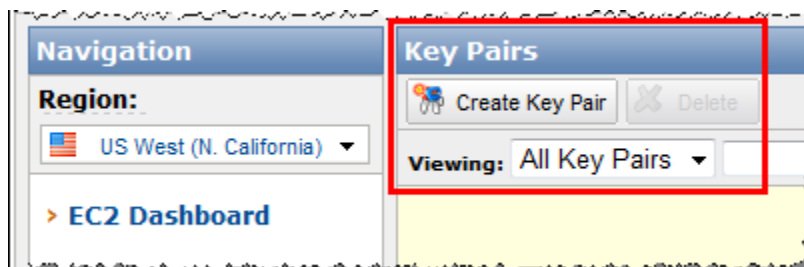
### To create an Amazon EC2 key pair

1. Sign in to the AWS Management Console and open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the Navigation pane, in the Region list, click the region in which you want to create the key pair.



You must create the key pair in the same region where you will create your AWS CloudFormation stack for live streaming later in this process. We recommend that you create the key pair and the stack for live streaming in the region that is closest to the users who will be doing the streaming.

3. In the Navigation pane, click **Key Pairs**.
4. In the Key Pairs pane, click **Create Key Pair**.



5. In the Create Key Pair dialog box, enter a name for the key pair, and make note of the name. You'll enter this value when you create an AWS CloudFormation live-streaming stack, later in the process of setting up live streaming.
6. Click **Create**.
7. In the Opening <key\_pair\_name>.pem dialog box, save the .pem file to a safe place on your computer.
8. Click **Close** to close the Create Key Pair dialog box.

Next: [Subscribing to Adobe Flash Media Server \(p. 155\)](#)

## Subscribing to Adobe Flash Media Server

Perform the following procedure to subscribe to Adobe Flash Media Server for Amazon Web Services with your AWS account.

There is a \$5.00 monthly subscription fee. This fee allows you to run an unlimited number of Flash Media Server instances. In addition to the monthly subscription fee, there is a fee for hourly usage and a fee for data transfer. For more information, see Amazon EC2 Pricing at <http://aws.amazon.com/ec2/pricing>.

### To order Adobe Flash Media Server for Amazon Web Services

1. Go to [http://www.adobe.com/go/learn\\_fms\\_aws\\_order\\_en](http://www.adobe.com/go/learn_fms_aws_order_en).
2. Follow the on-screen instructions.

#### Note

Read the product license agreement at <http://www.adobe.com/products/eulas>.

3. Read the pricing terms, and click **Place Your Order**.

The screenshot shows the Amazon Payments checkout page. At the top, it says 'Review the information below, then click "Place your order."' with a 'Place your order' button. Below this, there's a 'Total due today' section showing a one-time charge of \$5.00, a prorated recurring monthly charge of \$3.17, and a total of \$8.17. To the right, there's a 'Payment Method' section showing 'Visa' and an expiration date, and a 'Billing Address' section showing 'Jeff Barr'. Below these, there's an 'Application Information' section titled 'Adobe Flash Media Server on Amazon Web Services' with a description of the service. Underneath, there's a 'Pricing' section titled 'Adobe Flash Media Server on Amazon Web Services Pricing' which includes a table for 'One-Time and Monthly Charges' and a table for 'Usage Charges - US East (Northern Virginia) Region'.

Pricing Dimension	Your Price (\$)	Description
One-Time Charge	5.00	Adobe Flash Media Server Licensing Fee
Recurring Monthly Charge	5.00	Adobe Flash Media Server Monthly Licensing Fee

Instance	Your Price (\$)	Details	Description
Amazon EC2 running Linux/UNIX			
Large	0.440	per hour (or partial hour) consumed	Limited to 100 RTMPFP simultaneous connections
Extra Large	1.300	per hour (or partial hour)	Limited to 1,000 RTMPFP simultaneous

Next: [Creating an AWS CloudFormation Stack for Live Streaming](#) (p. 155)

## Creating an AWS CloudFormation Stack for Live Streaming

The following procedure uses an AWS CloudFormation template to create a stack that launches the AWS resources required by live streaming, including an Amazon EC2 instance.

## **Important**

You incur hourly charges for an Amazon EC2 instance beginning when you create the AWS CloudFormation stack that deploys the Amazon EC2 instance. Charges continue to accrue until you delete the AWS CloudFormation stack regardless of whether you use the Amazon EC2 instance to stream live video. For more information, see [Adobe Flash Media Server on Amazon Web Services Pricing](#). When your live event is over, delete the stack that you created for live streaming. This deletes the AWS resources that were created for your live-streaming event, and stops the AWS charges for the resources. For more information, see [Deleting an AWS CloudFormation Stack for Live Streaming \(p. 166\)](#).

## **To create an AWS CloudFormation stack for live streaming**

1. In the following list, click the Amazon EC2 Region where you want to create the stack. The Create Stack wizard starts, and a region-specific value is automatically entered in the **Provide a Template URL** field.

[US East \(Virginia\)](#)

[US West \(Oregon\)](#)

[US West \(Northern California\)](#)

[EU \(Ireland\)](#)

[Asia Pacific \(Singapore\)](#)

[Asia Pacific \(Tokyo\)](#)

[South America \(Sao Paulo\)](#)

## **Note**

If you want users to view your live stream using a Flash-based player that is hosted on your own domain, see [How do I update crossdomain.xml so my users can view the live stream using a Flash-based player that is hosted on my own domain? \(p. 168\)](#).

2. If you are not already signed in to the AWS Management Console, sign in when prompted.
3. *Optional:* Change the value of the **Stack Name** field. The stack name must not contain spaces, and it must be unique within your AWS account.



## Amazon CloudFront Developer Guide

### Creating an AWS CloudFormation Stack for Live Streaming

The screenshot shows the 'Create Stack' wizard in the AWS Management Console. The title bar says 'Create Stack' with a 'Cancel' button. Below the title bar is a progress bar with three steps: 'SELECT TEMPLATE' (active), 'SPECIFY PARAMETERS', and 'REVIEW'. The main text explains that AWS CloudFormation allows creating a collection of related AWS resources (a stack) by describing requirements in a template. It suggests using sample templates, uploading a file, or providing a URL. The 'Stack Name' field contains 'CompanyMeeting'. Under 'Stack Template Source', the 'Provide a Template URL' option is selected with a radio button. The URL field contains 'https://s3.amazonaws.com/cloudfront-live-us-w/live-http-s'. The 'Show Advanced Options' checkbox is checked. Under 'Notifications: (optional)', the 'Amazon SNS Topic' dropdown is set to '(no notification)'. The 'Creation Timeout (minutes)' dropdown is set to 'none'. The 'Rollback on failure' radio buttons are set to 'Yes'. A 'Continue' button with a right arrow is at the bottom.

**Create Stack** Cancel

**SELECT TEMPLATE** **SPECIFY PARAMETERS** **REVIEW**

AWS CloudFormation gives you an easier way to create a collection of related AWS resources (a stack) by describing your requirements in a template. To create a stack, fill in the name for your stack and select a template. You may choose one of the sample templates to get started quickly, or one of your own templates stored in S3 or on your local hard drive.

**Stack Name:**  
CompanyMeeting

**Stack Template Source:**

☐ Use a sample template

☐ Upload a Template File

☒ Provide a Template URL

https://s3.amazonaws.com/cloudfront-live-us-w/live-http-s

☒ **Show Advanced Options**

**Notifications: (optional)**

**Amazon SNS Topic** (no notification) ▼

**Creation Timeout (minutes):** none ▼

**Rollback on failure:** ☒ Yes ☐ No

**Continue** ▶

4. Do not change the **Stack Template Source** option or the value of **Provide a Template URL**.
5. *Optional:* To configure SNS notification, to specify how long you're willing to wait for the stack to be created, and to choose whether to roll back changes if stack creation fails, check the **Show Advanced Options** checkbox, and specify the applicable values.
6. Click **Continue**, and the Specify Parameters page of the Create Stack wizard appears.

**Create Stack**
Cancel

SELECT TEMPLATE
SPECIFY PARAMETERS
REVIEW

**Template Description:** This template creates a CloudFormation stack that uses Amazon CloudFront and an Amazon EC2 AMI for Adobe Flash Media Server 4.5 to enable HTTP streaming of your live event.

**Specify Parameters**

Below are the parameters associated with your CloudFormation template. You may review and proceed with the default parameters or make customizations as needed below.

**KeyPair**

The name of an Amazon EC2 key pair in the region in which you are creating a CloudFormation stack.

**FMSAdminPassword**

Enter a Password (minimum 8 characters) for the Flash Media Administration Console.

**FMSAdminUsername**

Enter a Username for the Flash Media Administration Console.

**StreamName**

A short name for your live stream (no spaces allowed)

**InstanceType**

The type of Amazon EC2 instance to launch.

< Back
Continue

7. In the **KeyPair** field, enter the name of an Amazon EC2 key pair in the region in which you want to create the stack for live streaming. The key pair must be associated with the account that you're currently logged on with. If you created a key pair when you performed the procedure in [Creating an Amazon EC2 Key Pair](#) (p. 154), enter the name of that key pair.
8. In the **StreamName** field, enter a short name (without spaces) for your live stream.
9. In the **FMSAdminUserName** field, enter the user name that you want to use to log on to the Flash Media Administration Console after your Amazon EC2 Flash Media Server instance is created.
10. In the **FMSAdminPassword** field, enter a password (minimum 8 characters) for the user name that you specified in Step 9.
11. In the **InstanceType** field, enter an instance type, and click **Continue**. The default value is *m1.xlarge*.

The instance type determines pricing for your Adobe Flash Media Server instance. The following table lists the Amazon EC2 instance types that are supported for live streaming.

For more information about Amazon EC2 instance types, see [Available Instance Types](#).

For information about pricing, go to the Adobe Flash Media Server on Amazon Web Services page, <http://www.adobe.com/ap/products/flashmediaserver/amazonwebservices/>, and click the **Pricing** tab.

Amazon EC2 and Adobe Flash Media Server Instance Types	InstanceType Code
Large	m1.large
High-memory extra large	m2.xlarge
High-CPU extra large	c1.xlarge

## Amazon CloudFront Developer Guide

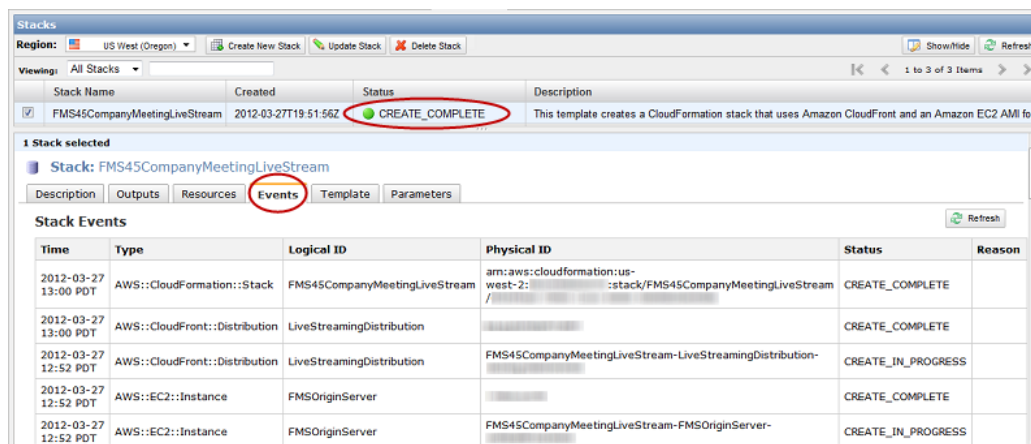
### Creating an AWS CloudFormation Stack for Live Streaming

Amazon EC2 and Adobe Flash Media Server Instance Types	InstanceType Code
Extra large	m1.xlarge
High-memory double extra large	m2.2xlarge
High-memory quadruple extra large	m2.4xlarge

12. Review the settings for the stack. When you're satisfied with the settings, click **Create Stack**.

Your stack may take several minutes to create. To track the progress of the stack creation, select the stack, and click the **Events** tab in the bottom frame. If AWS CloudFormation cannot create the stack, the Events tab lists error messages.

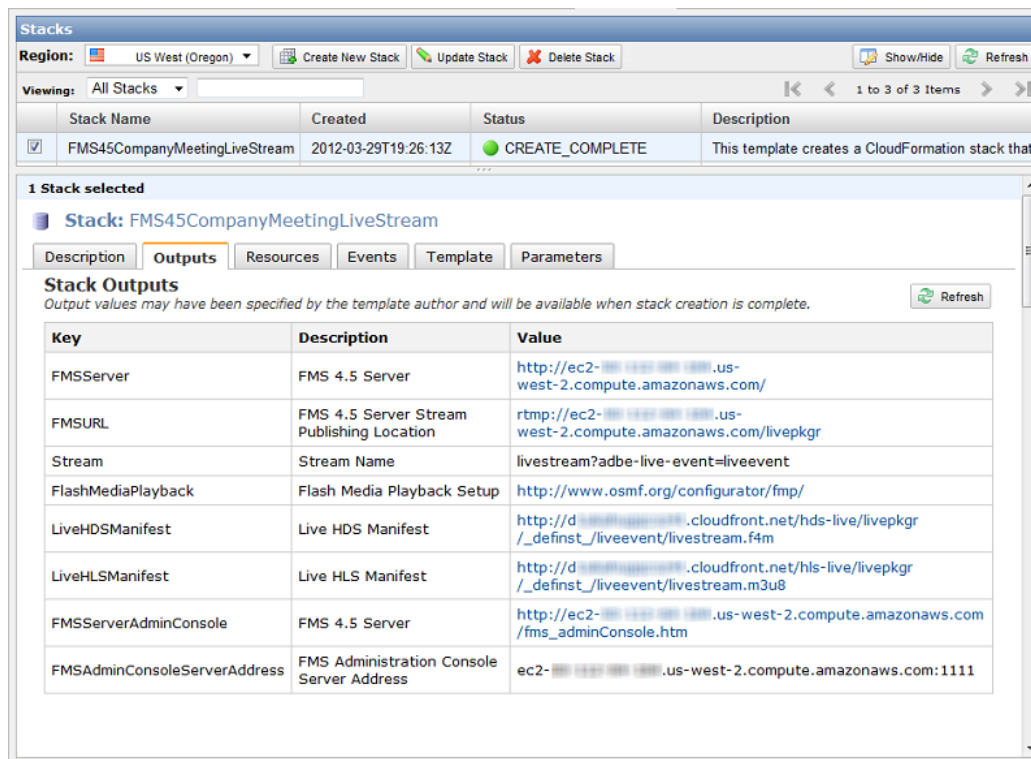
When your stack is ready, in the top frame, the status for the stack changes to **CREATE\_COMPLETE**.



The screenshot shows the AWS CloudFormation console. At the top, the 'Stacks' section is active, displaying a table of stacks. The stack 'FMS45CompanyMeetingLiveStream' is selected, and its status is 'CREATE\_COMPLETE'. Below this, the 'Stack: FMS45CompanyMeetingLiveStream' section is shown, with the 'Events' tab selected. The 'Stack Events' table lists the following events:

Time	Type	Logical ID	Physical ID	Status	Reason
2012-03-27 13:00 PDT	AWS::CloudFormation::Stack	FMS45CompanyMeetingLiveStream	arn:aws:cloudformation:us-west-2::stack/FMS45CompanyMeetingLiveStream	CREATE_COMPLETE	
2012-03-27 13:00 PDT	AWS::CloudFront::Distribution	LiveStreamingDistribution		CREATE_COMPLETE	
2012-03-27 12:52 PDT	AWS::CloudFront::Distribution	LiveStreamingDistribution	FMS45CompanyMeetingLiveStream-LiveStreamingDistribution-	CREATE_IN_PROGRESS	
2012-03-27 12:52 PDT	AWS::EC2::Instance	FMSOriginServer		CREATE_COMPLETE	
2012-03-27 12:52 PDT	AWS::EC2::Instance	FMSOriginServer	FMS45CompanyMeetingLiveStream-FMSOriginServer-	CREATE_IN_PROGRESS	

When your stack is created, click the **Outputs** tab, which displays the stack creation outputs. You will use these values when you set up Adobe Flash Media Live Encoder, later in the process.



Next: [Verifying that Adobe Flash Media Server Is Running \(p. 160\)](#)

## Verifying that Adobe Flash Media Server Is Running

After AWS CloudFormation creates the stack, perform the following procedure to verify that Adobe Flash Media Server is running on the Amazon Amazon EC2 instance that you provisioned by using AWS CloudFormation.

### To verify that Adobe Flash Media Server is running

1. Open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation/>.
2. In the top pane, select the stack that you created in [Creating an AWS CloudFormation Stack for Live Streaming \(p. 155\)](#).
3. In the bottom pane, click the **Outputs** tab.
4. Click on the value of the **FMSServer** key, for example, **<http://ec2-00-11-22-33.us-west-1.compute.amazonaws.com>**.
5. The Adobe Flash Media Server page appears and begins streaming content, which shows that Adobe Flash Media Server is running.

If streaming does not start, return to [Overview of Live HTTP Streaming with Amazon Web Services \(p. 152\)](#), and verify that the values you specified in the first five tasks are correct.

If the values were all correct, but streaming still has not started, see [How do I troubleshoot my Amazon EC2 instance if streaming doesn't start? \(p. 172\)](#).

Next: [Setting Up Adobe Flash Media Live Encoder to Publish a Live Stream \(p. 161\)](#)

## Setting Up Adobe Flash Media Live Encoder to Publish a Live Stream

Adobe Flash Media Server on Amazon Web Services includes an application called livepkgr that packages published streams for delivery using HTTP Dynamic Streaming (HDS) and HTTP Live Streaming (HLS).

The following procedure shows how to set up Adobe Flash Media Live Encoder (FMLE) to publish your live stream to the livepkgr application on Adobe Flash Media Server 4.5.

### Note

The Windows version of Flash Media Live Encoder doesn't support the AAC audio format. To add support for AAC, Adobe recommends that you purchase the [MainConcept AAC encoder](#).

### To specify live-streaming settings in Flash Media Live Encoder

1. Log on to the computer that you'll use to broadcast the live stream on the day of the event.
2. Open a web browser, and browse to the Adobe Flash Media Live Encoder page, <http://www.adobe.com/products/flashmediaserver/flashmediaencoder>.
3. Download and install the Flash Media Live Encoder.

### Note

Flash Media Live Encoder is free, but you need an Adobe account (also free) to download it.

4. Open the Flash Media Live Encoder `config.xml` file in a text editor. The default installation location is:

- **Windows:** `C:\Program Files\Adobe\Flash Media Live Encoder 3.2.`
- **Macintosh:** `Applications:Adobe:Flash Media Live Encoder 3.2.`

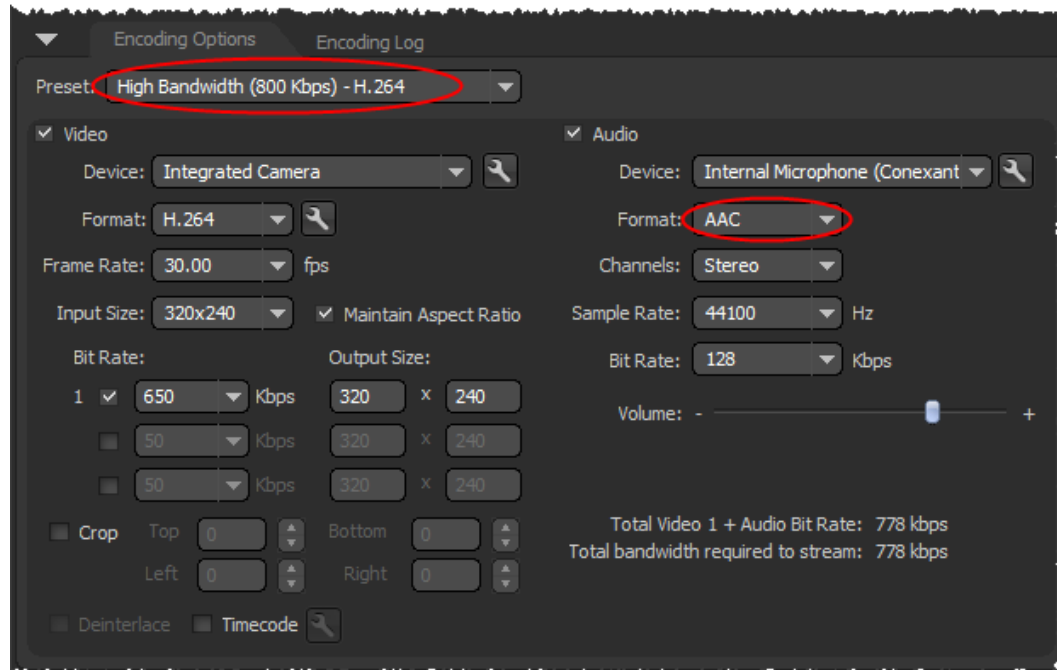
5. Set the value of the following `<enable>` element to `true`:

```
<flashmedialiveencoder_config>
...
  <mbrconfig>
    ...
    <streamsynchronization>
      ...
      <!-- "true" to enable this feature, "false" to disable. -->
      <enable>true</enable>
    
```

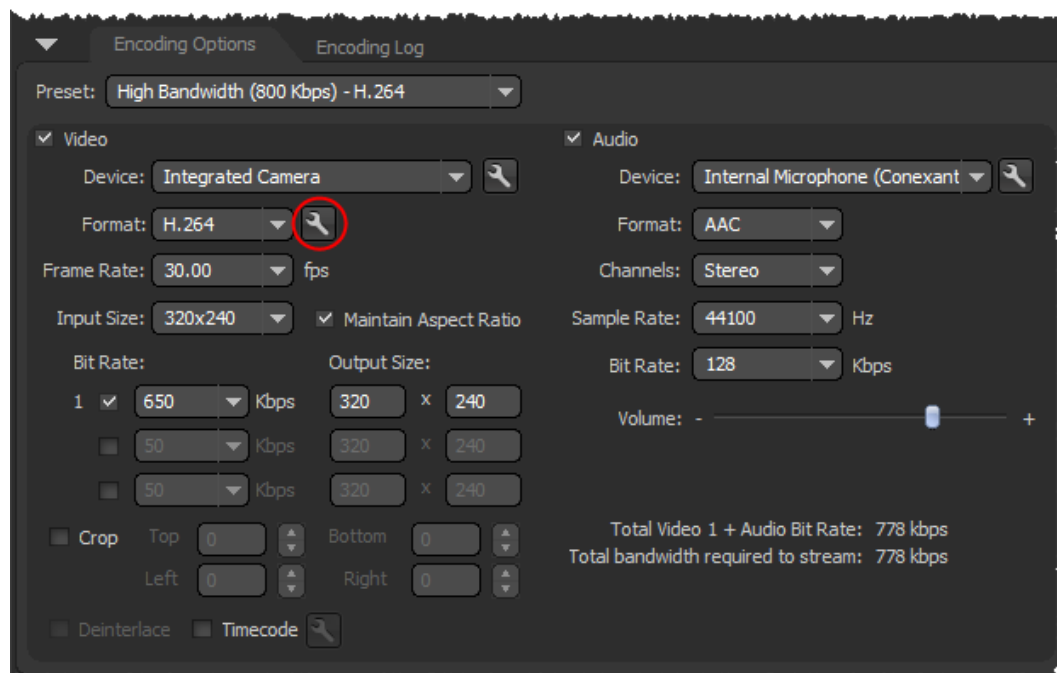
6. Save the file.
7. Run Flash Media Live Encoder.
8. On the **Encoding Options** tab, in the Preset list, click **High Bandwidth (800 Kbps) — H.264**.
9. On the **Encoding Options** tab, under the **Audio** check box, in the **Format** list, click **AAC**.

**Amazon CloudFront Developer Guide**  
**Setting Up Adobe Flash Media Live Encoder to Publish**  
**a Live Stream**

---

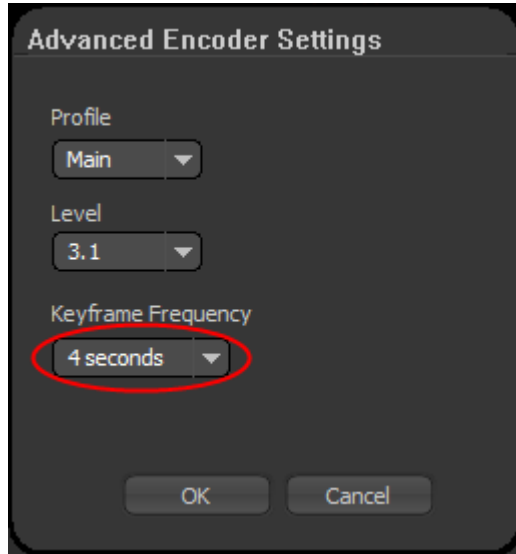


10. In the Video section of the Encoding Options tab, click the wrench icon on the right side of the Format list to open the Advanced Encoder Settings dialog box.



11. In the Advanced Encoder Settings dialog box, in the **Keyframe Frequency** list, click **4 Seconds**.

You can also use a multiple of the value of the <FragmentDuration> element in the applications/livepkgr/events/\_definst\_/liveevent/Event.xml file. The default value of <FragmentDuration> is 4000 milliseconds (4 seconds).



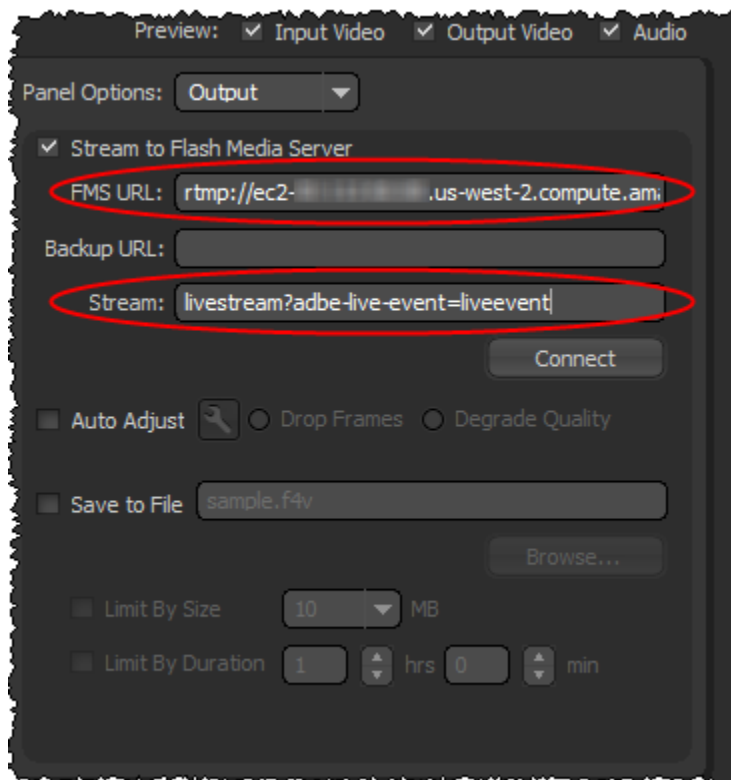
12. Click **OK** to save the setting and return to the main page. The value of the Preset list changes to Custom.
13. Open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation/>.
14. Check the checkbox for the stack that you created for live streaming.
15. In the bottom pane of the AWS CloudFormation console, click the **Outputs** tab.
16. Copy the value of the **FMSURL** key, for example, `rtmp://ec2-00-11-22-33.us-west-1.compute.amazonaws.com/livepkggr`.
17. In Flash Media Live Encoder, in the Stream to Flash Media Server section, in the **FMS URL** field, paste the value of the FMSURL key that you copied from the AWS CloudFormation console.
18. In the AWS CloudFormation console, copy the value of the **Stream** key, for example, `livestream?adbe-live-event=liveevent`.
19. In Flash Media Live Encoder, in the **Stream** field, paste the value of the **Stream** key that you copied from the AWS CloudFormation console.

#### **Note**

If you anticipate having to stop and restart the live stream, enter the following value in the **Stream** field instead:

```
livestream?adbe-live-event=liveevent&adbe-record-mode=record
```

If you publish a live stream in record mode (`adbe-record-mode=record`), then stop the stream and restart it, Adobe Flash Media Server will delete the previous stream and start a new stream instead of appending to the previous stream when you restart. However, if you don't use record mode and you stop the live stream, you have to reconfigure live streaming before you can restart the stream.



20. Uncheck the **Save to File** checkbox.
21. Click **Connect** to connect to your Flash Media Server instance.
22. Click **Start** to start encoding and publishing your live stream to the livepkgr application on your Adobe Flash Media Server instance.

Next: [Embedding Flash Media Playback for an Amazon CloudFront Live HTTP Stream in a Web Application](#) (p. 164)

## Embedding Flash Media Playback for an Amazon CloudFront Live HTTP Stream in a Web Application

Perform the applicable procedure to get the embed code that you will include in your web page for the live stream:

- [To embed Flash Media playback for your HTTP stream via CloudFront](#) (p. 164)
- [To play your live HLS stream on an Apple device via CloudFront](#) (p. 166)

### To embed Flash Media playback for your HTTP stream via CloudFront

1. Open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation/>.
2. Check the checkbox for the stack for live streaming.
3. In the bottom pane of the AWS CloudFormation console, click the **Outputs** tab.
4. Copy the value of the **LiveHDSManifest** key, for example, `http://d123.cloudfront.net/hds-live/livepkgr/_definst_/liveevent/livestream.f4m`.
5. Click the value of the **FlashMediaPlayback** key to open the Flash Media Playback Setup webpage.





```
<object width="600" height="409"><param name="movie" value="http://fpdownload.adobe.com/strobe/FlashMediaPlayback_101.swf"></param><param name="flashvars" value="src=http%3A%2F%2Fdd.cloudflare.net%2Fstatic-live%2Flivepgkr%2F_definst_%2Fliveevent%2Flivestream.f4m"></param><param name="allowFullScreen" value="true"></param><param name="allowscriptaccess" value="always"></param><embed src="http://fpdownload.adobe.com/strobe/FlashMediaPlayback_101.swf" type="application/x-shockwave-flash" allowscriptaccess="always" allowfullscreen="true" width="600" height="409" flashvars="src=http%3A%2F%2Fdd.cloudflare.net%2Fstatic-live%2Flivepgkr%2F_definst_%2Fliveevent%2Flivestream.f4m"></embed></object>
```

- ## Note

## To play your live HLS stream on an Apple device via CloudFront

- For information about where to use the URL to serve various iOS devices, QuickTime, and Safari, go to [HTTP Live Streaming Overview](#) in the iOS Developer Library.

Next: [Deleting an AWS CloudFormation Stack for Live Streaming \(p. 166\)](#)

When your live event is over, delete the stack that you created for live streaming. This deletes the AWS resources that were created for your live-streaming event, and stops the AWS charges for the resources.

1. Sign in to the AWS Management Console and open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation/>.
2. Check the checkbox for the stack, and click **Delete Stack**.
3. Click **Yes, Delete** to confirm.
4. To track the progress of the stack deletion, check the checkbox for the stack, and click the **Events** tab in the bottom frame.

5. If you do not plan to use live streaming again soon, you can cancel your subscription to Adobe Flash Media Server on Amazon EC2. To cancel the subscription, go to [http://www.adobe.com/go/learn\\_fms\\_aws\\_order\\_en](http://www.adobe.com/go/learn_fms_aws_order_en), and follow the on-screen prompts.

## Frequently Asked Questions

- How can I use Secure Shell (SSH) to connect to my Amazon EC2 instance that is running Adobe Flash Media Server 4.5? (p. 167)
- How do I update `crossdomain.xml` so my users can view the live stream using a Flash-based player that is hosted on my own domain? (p. 168)
- What is the price for live HTTP streaming using CloudFront and Adobe Flash Media Server 4.5? (p. 169)
- How can I create a CNAME alias for my Amazon EC2 instance or for my CloudFront distribution? (p. 169)
- How can I connect to the Flash Media Administration Console? (p. 169)
- Can I stream my live event both to Apple devices and to Flash Player-compatible devices? (p. 170)
- Does Flash Media Server 4.5 support HTML5? (p. 171)
- Does Flash Media Server have logging? (p. 171)
- How can I enable authentication on Flash Media Server? (p. 171)
- What are the default cache-control settings on HDS- and HLS-related files? (p. 171)
- What is the difference between HLS and HDS? (p. 171)
- How do I troubleshoot my Amazon EC2 instance if streaming doesn't start? (p. 172)
- Where can I find the documentation for live streaming using Adobe Flash Media Server 4.0? (p. 172)

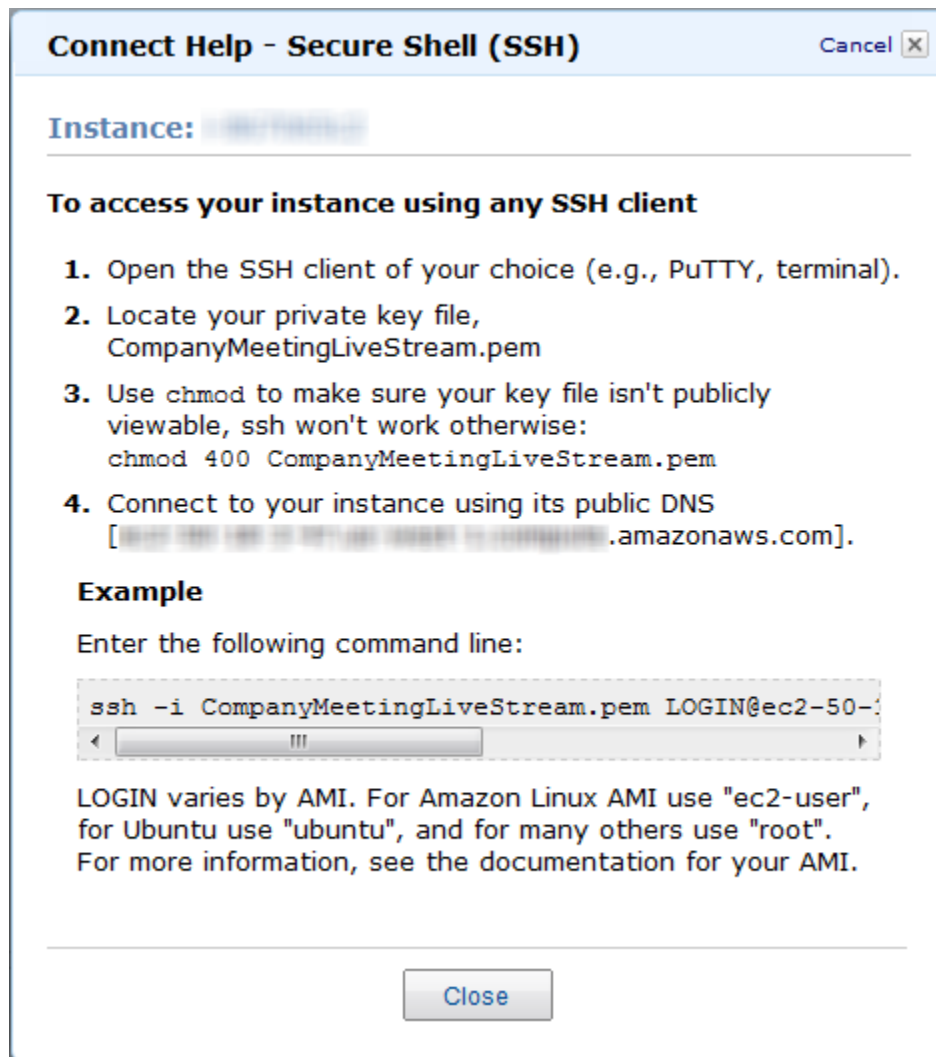
## How can I use Secure Shell (SSH) to connect to my Amazon EC2 instance that is running Adobe Flash Media Server 4.5?

### Note

By default, the SSH port for the Amazon EC2 instance (port 22) is disabled for security reasons. The following procedure explains how to enable the SSH port and how to use SSH to connect to your Amazon EC2 instance.

### To use SSH to connect to your Amazon EC2 instance that is running Adobe Flash Media Server 4.5

1. Sign in to the AWS Management Console.
2. Authorize network access to your Amazon EC2 instance. For more information, see [Authorize Network Access to Your Instances](#) in the *Amazon Elastic Compute Cloud User Guide*.
3. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
4. In the Navigation pane, click **Instances**.
5. Right-click the correct instance, and click **Connect** to view instructions on how to use SSH to connect to your Amazon EC2 instance:



## How do I update crossdomain.xml so my users can view the live stream using a Flash-based player that is hosted on my own domain?

You can change permissions in `crossdomain.xml` either before or after you create the AWS CloudFormation stack:

- If you have not created your AWS CloudFormation stack, download the AWS CloudFormation template for Live Streaming using Amazon CloudFront and Adobe Flash Media Server 4.5 at <https://cloudfront-live.s3.amazonaws.com/live-http-streaming-fms-4-5-1-using-cloudfront.txt>. In the template, edit the `UserData` section, which contains the `crossdomain.xml` settings, and save the updated template on your local computer. Then create your AWS CloudFormation stack using the updated template.
- If you have already created your AWS CloudFormation stack, log into Adobe Flash Media Server running on your Amazon EC2 instance, and change permissions in the cross-domain policy file, `/mnt/webroot/crossdomain.xml`.

For more information about editing the `crossdomain.xml` file, go to [Adobe Cross Domain Policy File Specification](#).

## What is the price for live HTTP streaming using CloudFront and Adobe Flash Media Server 4.5?

In addition to the \$5.00 monthly subscription fee for Adobe Flash Media Server on Amazon EC2, you pay only for the AWS resources you consume:

- For pricing information about Adobe Flash Media Server running on Amazon EC2, see the **Pricing** tab on the Adobe Flash Media Server on Amazon Web Services web page at <http://www.adobe.com/ap/products/flashmediaserver/amazonwebservices/>.
- For pricing information about CloudFront, see <http://aws.amazon.com/cloudfront/pricing>.

There is no charge for using AWS CloudFormation.

## How can I create a CNAME alias for my Amazon EC2 instance or for my CloudFront distribution?

Your Amazon EC2 instance running Adobe Flash Media Server 4.5 comes with an internal and an external DNS name. Amazon EC2 does not provide access to modify these DNS settings. If you want to map an existing domain name to your Amazon EC2 instance running Flash Media Server, use a DNS service provider such as [Amazon Route 53](#). When using your own domain name, we recommend that you map to the instance's external DNS name using a CNAME, not by using an A record that points to the instance's IP address.

To map your own domain name to your CloudFront distribution, see [Using Alternate Domain Names \(CNAMEs\)](#) (p. 53).

## How can I connect to the Flash Media Administration Console?

### To connect to the Flash Media Administration Console

1. Sign in to the AWS Management Console and open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation/>.
2. Select the stack for live streaming.
3. In the bottom pane of the AWS CloudFormation console, click the **Outputs** tab.
4. Click the value of the **FMSServerAdminConsole** key, for example, **http://ec2-00-11-22-33.us-west-1.compute.amazonaws.com/fms\_adminConsole.htm**.
5. In the AWS CloudFormation console, copy the value of the **FMSAdminConsoleServerAddress** key to the clipboard.
6. In the **Server Address** field, paste the value that you copied in the previous step.
7. In the **Username** and **Password** fields, enter the values that you specified in [Creating an AWS CloudFormation Stack for Live Streaming](#) (p. 155).
8. Click **Login**.

For information about using the Flash Media Server 4.5 Administration Console, refer to the [Adobe documentation](#).

## Note

Adobe recommends that you block all external access to port 1111 so that access to the Administration Console is restricted only to clients that are within your firewall. As an alternative, you can restrict access to the server by using domain-based restrictions. For more information, go to the [Limit access to Flash Media Administration Server](#) in the Adobe documentation.

### To disable or restrict access to port 1111 on your Flash Media Server

1. Get the name of the Amazon EC2 security group that is associated with your Amazon EC2 instance:
  - a. Sign in to the AWS Management Console and open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation/>.
  - b. In the Region list, click the name of the region in which you created your Amazon EC2 instance.
  - c. Click the row for your AWS CloudFormation stack.
  - d. In the bottom pane, click the **Resources** tab.
  - e. In the left column of the **Stack Resources** table, find the row for which the value is `FMSOriginServerSecurityGroup`.
  - f. For that row, write down the value of the **Physical ID** column.
2. Display the Amazon EC2 console.
3. In the **Navigation** pane, click **Security Groups**.
4. In the **Security Groups** pane, click the row for which the value of the **Name** column matches the value that you got in Step 1f.
5. In the bottom pane, click the **Inbound** tab.
6. To completely disable access to the Flash Media Administration console:
  - a. In the **TCP Port (Service)** column, find **1111**.
  - b. In the **Action** column for that row, click **Delete**.
  - c. Click **Apply Rule Changes**.

To restrict access to selected IP addresses:

- a. In the **TCP Port (Service)** column, find **1111**.
- b. In the **Action** column for that row, click **Delete**.
- c. In the **Create a new rule** list, do not change the default value, **Custom TCP rule**.
- d. In the **Port range** field, enter **1111**.
- e. In the **Source** field, enter an IP address or range, or enter the name of another security group. For more information, click **Help**.
- f. Click **Add Rule**.
- g. To create additional rules, repeat Steps c through f.
- h. Click **Apply Rule Changes**.

## Can I stream my live event both to Apple devices and to Flash Player-compatible devices?

Yes, Flash Media Server 4.5 enables the delivery of live streams to both Flash-based and iOS devices at the same time. You can stream to the Safari browser using an HTML5 player or an Objective C ("native") application. You can also use Adobe AIR for iOS to develop a rich video experience on iOS.

## Does Flash Media Server 4.5 support HTML5?

Yes. Flash Media Server can deliver content to HTML5 on Apple iOS devices using the HLS streaming format. For other browsers supporting HTML5, you can use Flash Media Server to deliver progressively.

## Does Flash Media Server have logging?

Yes. W3C-compliant ASCII logs, a real-time usage monitor, and a complete API for server and stream events help to ensure that you have all the tools you need to track and generate reports on your audience's content use. For more information about monitoring and managing log files in Flash Media Server 4.5, go to [Monitoring and Managing Log Files](#) in the Adobe documentation.

## How can I enable authentication on Flash Media Server?

You can download the [Authentication add-in](#) and install it on your Flash Media Server instance. You can also restrict access to RTMP port 1935 (for both TCP and UDP) in the security group created by AWS CloudFormation for your Flash Media Server Amazon EC2 instance. Just create new TCP and UDP rules for port 1935 and then delete the existing TCP and UDP rules for port 1935, which allow access to all IP addresses.

For a quick overview of how to add a rule to a security group, see [How can I connect to the Flash Media Administration Console?](#) (p. 169). For more information about Amazon EC2 security groups, go to the [Using Security Groups](#) in the *Amazon Elastic Compute Cloud User Guide*.

## What are the default cache-control settings on HDS- and HLS-related files?

The default cache control headers on HDS- and HLS-related files are set to the following values:

File Type	Cache-Control Setting (Seconds)
.bootstrap	2
.f4f	none (the CloudFront default value is 86400 seconds, or one day)
.f4m	2
.m3u8	2
.ts	none (the CloudFront default value is 86400 seconds, or one day)

The CloudFront edge cache servers honor these cache control headers. You can change the default settings by changing the values of the `HttpStreamingF4MMaxAge`, `HttpStreamingBootstrapMaxAge`, and `HttpStreamingFragMaxAge` parameters on the server. For more information, go to [HTTP streaming configuration file reference](#) on the Adobe website.

## What is the difference between HLS and HDS?

HLS is a file container format optimized for Apple devices. The container supports H.264/AAC-encoded video and audio, and is based on MPEG-2 transport stream (TS). All video delivered to iOS (including AIR for iOS) must use this format.

HDS is a file container format optimized for applications that run in Flash Player. The container also supports H.264/AAC-encoded video and audio and is based on MPEG-4 TS. HDS is not supported on AIR for iOS.

## How do I troubleshoot my Amazon EC2 instance if streaming doesn't start?

If you performed the procedure [To verify that Adobe Flash Media Server is running \(p. 160\)](#) and streaming still hasn't started, perform the following procedure to confirm that the Amazon EC2 instance is functioning correctly.

### To troubleshoot your Amazon EC2 instance running Adobe Flash Media Server 4.5

1. In the AWS CloudFormation console, in the top pane, select the stack.
2. In the bottom pane, click the **Resources** tab.
3. For the **FMSOriginServer** row, write down the value of the **Physical ID** column.
4. Go to the Amazon EC2 console.
5. In the **Navigation** pane, select the region in which you created the AWS CloudFormation stack.
6. In the **Navigation** pane, click **Instances**.
7. In the **Instance** column, find the value that you wrote down in Step c.
8. Select the corresponding row.
9. In the bottom pane, review the information on the **Status Checks** tab, and take the recommended actions.
10. Return to the procedure [To verify that Adobe Flash Media Server is running \(p. 160\)](#), and repeat Steps 2 through 5.

## Where can I find the documentation for live streaming using Adobe Flash Media Server 4.0?

For the documentation for the original release of live streaming, using Adobe Flash Media Server 4.0, see [Live Streaming Using CloudFront and Adobe Flash Media Server](#).

## Additional Documentation

### Adobe Documentation

- Using Adobe Flash Media Server on Amazon Web Services,  
<http://www.adobe.com/products/flashmediaserver/amazonwebservices/>
- Adobe Cross Domain Policy File Specification,  
[http://learn.adobe.com/wiki/download/attachments/64389123/CrossDomain\\_PolicyFile\\_Specification.pdf](http://learn.adobe.com/wiki/download/attachments/64389123/CrossDomain_PolicyFile_Specification.pdf)
- Adobe Flash media Live Encoder:
  - <http://www.adobe.com/products/flashmediaserver/flashmediaencoder>
  - <http://www.adobe.com/products/flash-media-encoder/faq.html>
- Video: Encoding and Transcoding Recommendations for HTTP Dynamic Streaming on the Adobe Flash Platform, [http://download.macromedia.com/flashmediaserver/http\\_encoding\\_recommendations.pdf](http://download.macromedia.com/flashmediaserver/http_encoding_recommendations.pdf)
- *Adobe Flash Media Server 4.5 Technical Overview*,  
[http://help.adobe.com/en\\_US/flashmediaserver/techoverview/flashmediaserver\\_4.5\\_tech\\_overview.pdf](http://help.adobe.com/en_US/flashmediaserver/techoverview/flashmediaserver_4.5_tech_overview.pdf).



## Amazon Web Services Documentation

- Amazon Elastic Compute Cloud, <http://aws.amazon.com/documentation/ec2>
- CloudFront, <http://aws.amazon.com/documentation/cloudfront>
- AWS CloudFormation, <http://aws.amazon.com/documentation/cloudformation>

# Live Smooth Streaming Using Amazon CloudFront and IIS Media Services 4.1

## Topics

- [Overview of Live Smooth Streaming with Amazon Web Services \(p. 174\)](#)
- [Creating an Amazon Web Services Account \(p. 175\)](#)
- [Creating an Amazon EC2 Key Pair \(p. 175\)](#)
- [Creating an AWS CloudFormation Stack for Live Smooth Streaming \(p. 176\)](#)
- [Verifying that Your Amazon EC2 Windows Server Instance Is Running \(p. 179\)](#)
- [Getting Your Windows Password \(p. 180\)](#)
- [Encoding Your Live Stream \(p. 182\)](#)
- [Viewing Your Live Smooth Stream \(p. 187\)](#)
- [Deleting Your AWS CloudFormation Live Smooth Streaming Stack \(p. 187\)](#)
- [Frequently Asked Questions \(p. 188\)](#)
- [Additional Documentation \(p. 190\)](#)

## Overview of Live Smooth Streaming with Amazon Web Services

Smooth Streaming is the Microsoft implementation of adaptive streaming technology, which is a form of web-based media content delivery that uses standard HTTP. An extension of IIS Media Services, Smooth Streaming enables adaptive streaming of live events to Smooth Streaming clients such as Microsoft Silverlight. When you configure Smooth Streaming to use CloudFront, you benefit from the scale of CloudFront's global HTTP network and from latency-based routing of viewers to edge nodes on the network. To learn more about CloudFront, go to the [CloudFront product page](#).

Smooth Streaming content is delivered to clients as a series of MPEG-4 (MP4) fragments that can be cached at the CloudFront edge servers. Smooth Streaming-compatible clients use special heuristics to dynamically monitor current network and local PC conditions, and seamlessly switch the video quality of the Smooth Streaming presentation that the clients receive. As clients play the fragments, network conditions may change (for example, bandwidth may decrease) or video processing may be affected by other applications that are running on the client. Clients can immediately request that the next fragment come from a stream that is encoded at a different bit rate to accommodate the changing conditions. This enables clients to play the media without stuttering, buffering, or freezing. As a result, users experience the highest-quality playback available without interruptions in the stream.

To encode a live broadcast to Smooth Streaming format, you use Microsoft Expression Encoder 4 Pro. To serve the encoded Smooth Stream, you can then use an Amazon EC2 Amazon Machine Image (AMI) that is running Windows IIS Media Services. CloudFront caches the live video and audio content, and viewers connect to the CloudFront edge servers to play the stream using a Smooth Streaming-compatible client such as Microsoft Silverlight. This tutorial walks you through the entire setup process.

To set up Live Smooth Streaming with Amazon Web Services (AWS), review the system requirements for IIS Smooth Streaming in the [Smooth Streaming Deployment Guide](#). Then perform the procedures in the following sections:

1. [Creating an Amazon Web Services Account \(p. 175\)](#)
2. [Creating an Amazon EC2 Key Pair \(p. 175\)](#)
3. [Creating an AWS CloudFormation Stack for Live Smooth Streaming \(p. 176\)](#)
4. [Verifying that Your Amazon EC2 Windows Server Instance Is Running \(p. 179\)](#)

5. [Getting Your Windows Password](#) (p. 180)
6. [Encoding Your Live Stream](#) (p. 182)
7. [Viewing Your Live Smooth Stream](#) (p. 187)
8. [Deleting Your AWS CloudFormation Live Smooth Streaming Stack](#) (p. 187)

For frequently asked questions, see [Frequently Asked Questions](#) (p. 188).

For links to additional Microsoft and AWS documentation, see [Additional Documentation](#) (p. 190).

## Creating an Amazon Web Services Account

If you already have an AWS account, skip to [Creating an Amazon EC2 Key Pair](#) (p. 175). If you don't already have an AWS account, use the following procedure to create one.

### Note

When you create an account, AWS automatically signs up the account for all services. You are charged only for the services you use.

### To create an AWS account

1. Go to <http://aws.amazon.com>, and click **Create an AWS Account**.
2. Follow the on-screen instructions.

Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

Next: [Creating an Amazon EC2 Key Pair](#) (p. 175)

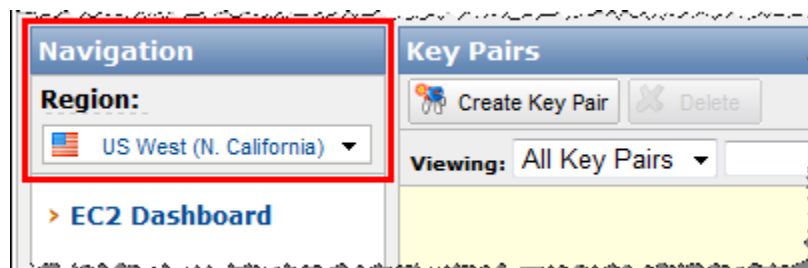
## Creating an Amazon EC2 Key Pair

If you already have an Amazon EC2 key pair in the Amazon EC2 region in which you want to configure Live Smooth Streaming, skip to [Creating an AWS CloudFormation Stack for Live Smooth Streaming](#) (p. 176). If you don't have a key pair in that region, perform the following procedure.

A key pair is a security credential similar to a password. You specify a key pair when you create an AWS CloudFormation stack for live streaming, later in this process. After live streaming is configured, you use the key pair to retrieve the password for your Amazon EC2 Windows Server instance.

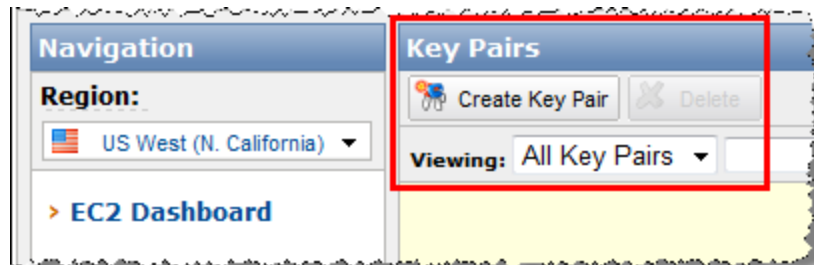
### To create an Amazon EC2 key pair

1. Sign in to the AWS Management Console and open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the Navigation pane, in the Region list, click the region in which you want to create the key pair.



You must create the key pair in the same region where you will create your AWS CloudFormation stack for live streaming later in this process. We recommend that you create the key pair and the stack for live streaming in the region that is closest to the location of your live event.

3. In the Navigation pane, click **Key Pairs**.
4. In the Key Pairs pane, click **Create Key Pair**.



5. In the Create Key Pair dialog box, enter a name for the key pair, and make note of the name. You'll enter this value when you create an AWS CloudFormation live-streaming stack, later in the process of setting up live streaming.
6. Click **Create**, and the Opening <key\_pair\_name>.pem dialog box appears.
7. Save the .pem file to a safe place on your computer.
8. Click **Close** to close the Create Key Pair dialog box.

Next: [Creating an AWS CloudFormation Stack for Live Smooth Streaming \(p. 176\)](#)

## Creating an AWS CloudFormation Stack for Live Smooth Streaming

The following procedure uses an AWS CloudFormation template to create a stack that launches the AWS resources required for Live Smooth Streaming, including an Amazon EC2 instance.

### Important

You incur hourly charges for an Amazon EC2 instance beginning when you create the AWS CloudFormation stack that deploys the Amazon EC2 instance. Charges continue to accrue until you delete the AWS CloudFormation stack regardless of whether you use the Amazon EC2 instance to stream live video. For more information, see [Pricing](#) on the Amazon Elastic Compute Cloud (Amazon EC2) detail page. When your live event is over, delete the stack that you created for Live Smooth Streaming. This deletes the AWS resources that were created for your live-streaming event, and stops the AWS charges for the resources. For more information, see [Deleting Your AWS CloudFormation Live Smooth Streaming Stack \(p. 187\)](#).

### To create an AWS CloudFormation stack for live streaming

1. In the following list, click the Amazon EC2 Region where you want to create the stack. The Create Stack wizard starts, and a region-specific value is automatically entered in the **Provide a Template URL** field.

[US East \(Virginia\)](#)

[US West \(Oregon\)](#)

[US West \(Northern California\)](#)

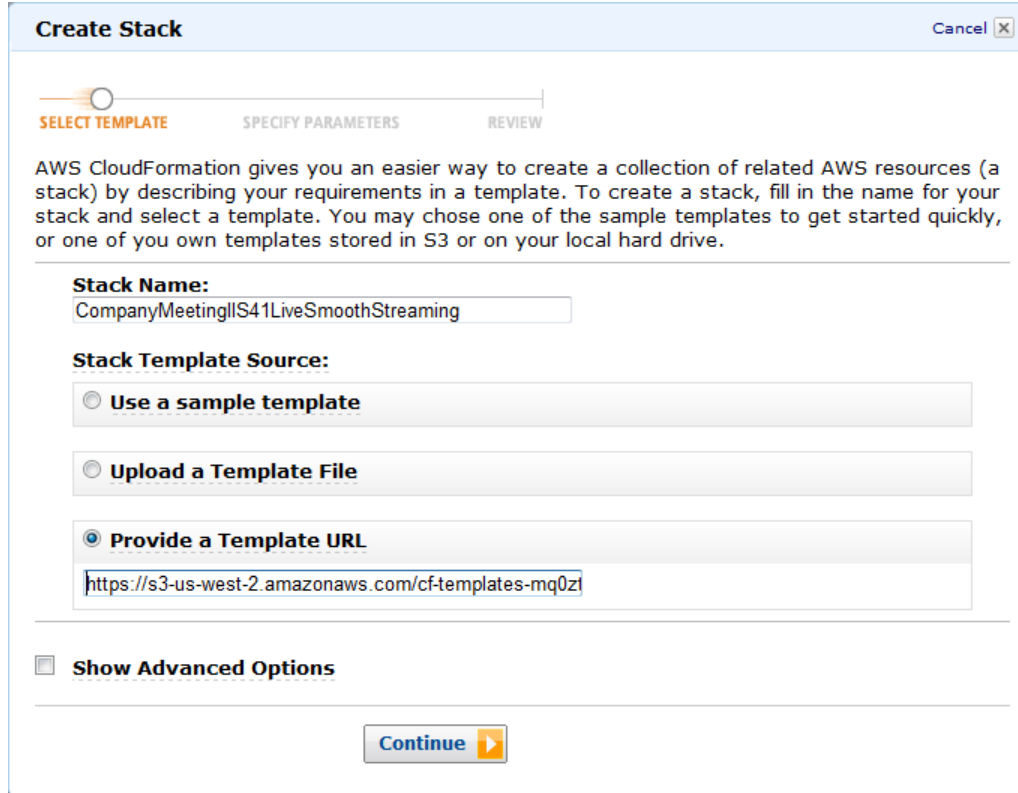
[EU \(Ireland\)](#)

[Asia Pacific \(Singapore\)](#)

[Asia Pacific \(Tokyo\)](#)

[South America \(Sao Paulo\)](#)

2. If you are not already signed in to the AWS Management Console, sign in when prompted.
3. *Optional:* In the Create Stack wizard, change the value of the **Stack Name** field. The stack name must not contain spaces, and it must be unique within your AWS account.



**Create Stack** Cancel

SELECT TEMPLATE | SPECIFY PARAMETERS | REVIEW

AWS CloudFormation gives you an easier way to create a collection of related AWS resources (a stack) by describing your requirements in a template. To create a stack, fill in the name for your stack and select a template. You may choose one of the sample templates to get started quickly, or one of your own templates stored in S3 or on your local hard drive.

**Stack Name:**  
CompanyMeetingIS41LiveSmoothStreaming

**Stack Template Source:**

☐ Use a sample template

☐ Upload a Template File

☒ Provide a Template URL  
`https://s3-us-west-2.amazonaws.com/cf-templates-mq0zi`

☐ Show Advanced Options

**Continue**

4. Do not change the **Stack Template Source** option or the value of **Provide a Template URL**.
5. *Optional:* To configure SNS notification, to specify how long you're willing to wait for the stack to be created, and to choose whether to roll back changes if stack creation fails, check the **Show Advanced Options** checkbox, and specify the applicable values.
6. Click **Continue**.

## Amazon CloudFront Developer Guide

### Creating an AWS CloudFormation Stack for Live Smooth Streaming

**Create Stack**Cancel

SELECT TEMPLATE

SPECIFY PARAMETERS

REVIEW

**Template Description:** This template uses Amazon CloudFront and the Windows IIS Media Services AMI to create a CloudFormation stack for Smooth streaming of your live event.

**Specify Parameters**

Below are the parameters associated with your CloudFormation template. You may review and proceed with the default parameters or make customizations as needed below.

**KeyPair**

IIS41LiveSmoothStreaming

The key pair name for your IIS Media Services EC2 instance

**InstanceType**

m1.xlarge

Type of EC2 instance to launch (default option is m1.large)

< BackContinue >

- On the Specify Parameters page, in the **KeyPair** field, enter the name of an Amazon EC2 key pair in the region in which you want to create the stack for live streaming. The key pair must be associated with the account that you're currently logged on with. If you created a key pair when you performed the procedure in [Creating an Amazon EC2 Key Pair \(p. 175\)](#), enter the name of that key pair.
- In the **InstanceType** field, enter an instance type, and click **Continue**. The default value is *m1.xlarge*.

The instance type determines the pricing for your Amazon EC2 instance that is running Windows Server. For more information about Amazon EC2 instance types for Windows, including pricing information, go to [Amazon EC2 Running Microsoft Windows Server & SQL Server](#).

- Review the settings for the stack. When you're satisfied with the settings, click **Create Stack**.

Your stack may take several minutes to create. To track the progress of the stack creation, select the stack, and click the **Events** tab in the bottom frame. If AWS CloudFormation cannot create the stack, the Events tab lists error messages.

When your stack is ready, in the top frame, the status for the stack changes to **CREATE\_COMPLETE**.

Stacks

Region: US West (Oregon) Create New Stack Update Stack Delete Stack Show/Hide Refresh

Viewing: All Stacks

Stack Name	Created	Status	Description
<input checked="" type="checkbox"/> CompanyMeetingIIS41LiveSmoothStreaming	2012-03-30T20:28:33Z	<span>CREATE_COMPLETE</span>	This template uses Amazon CloudFront and the Windows IIS Media Services AMI to create a CloudFormation stack for S

1 Stack selected

Stack: CompanyMeetingIIS41LiveSmoothStreaming

Description Outputs Resources **Events** Template Parameters

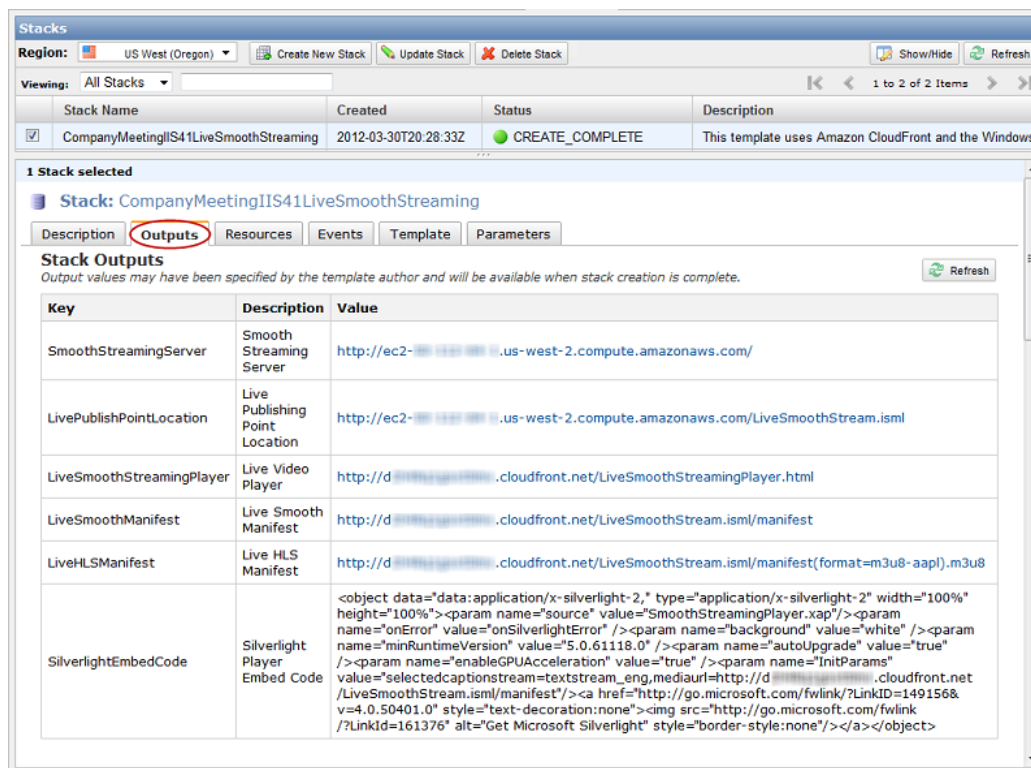
Stack Events

Time	Type	Logical ID	Physical ID	Status	Reason
2012-03-30 13:38 PDT	AWS::CloudFormation::Stack	CompanyMeetingIIS41LiveSmoothStreaming	arn:aws:cloudformation:us-west-2::stack/CompanyMeetingIIS41LiveSmoothStreaming	CREATE_COMPLETE	
2012-03-30 13:38 PDT	AWS::CloudFront::Distribution	SmoothStreamingDistribution		CREATE_COMPLETE	
2012-03-30 13:29 PDT	AWS::CloudFront::Distribution	SmoothStreamingDistribution	CompanyMeetingIIS41LiveSmoothStreaming-SmoothStreamingDistribution-	CREATE_IN_PROGRESS	
2012-03-30 13:29 PDT	AWS::EC2::Instance	SmoothStreamingOriginServer		CREATE_COMPLETE	
2012-03-30 13:28 PDT	AWS::EC2::Instance	SmoothStreamingOriginServer	CompanyMeetingIIS41LiveSm-SmoothStreamingOriginSer-	CREATE_IN_PROGRESS	
2012-03-30 13:28 PDT	AWS::EC2::SecurityGroup	SmoothStreamingServerSecurityGroup	CompanyMeetingIIS41LiveSmoothStreaming-SmoothStreamingServerSecurityGroup-	CREATE_COMPLETE	
2012-03-30 13:28 PDT	AWS::EC2::SecurityGroup	SmoothStreamingServerSecurityGroup	CompanyMeetingIIS41LiveSmoothStreaming-SmoothStreamingServerSecurityGroup-	CREATE_IN_PROGRESS	
2012-03-30 13:28 PDT	AWS::CloudFormation::Stack	CompanyMeetingIIS41LiveSmoothStreaming	arn:aws:cloudformation:us-west-2::stack/CompanyMeetingIIS41LiveSmoothStreaming	CREATE_IN_PROGRESS	User Initiated

## Amazon CloudFront Developer Guide

### Verifying that Your Amazon EC2 Windows Server Instance Is Running

When your stack is created, click the **Outputs** tab, which displays the stack creation outputs. You will use these values when you set up Microsoft Expression Encoder later in the process.



Next: [Verifying that Your Amazon EC2 Windows Server Instance Is Running \(p. 179\)](#)

## Verifying that Your Amazon EC2 Windows Server Instance Is Running

After AWS CloudFormation creates the stack, perform the following procedure to verify that your Windows IIS Media Services webserver is running on the Amazon EC2 instance that you provisioned via AWS CloudFormation.

### To verify that your Windows Server is running

1. Sign in to the AWS Management Console and open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation/>.
2. In the top pane, select the stack that you created in [Creating an AWS CloudFormation Stack for Live Smooth Streaming \(p. 176\)](#).
3. In the bottom pane, click the **Outputs** tab.
4. Click on the value of the **SmoothStreamingServer** key, for example, **<http://ec2-00-11-22-33.us-west-1.compute.amazonaws.com>**.

The Windows IIS Server banner screen appears, indicating that your Windows Server is running.

Next: [Getting Your Windows Password \(p. 180\)](#)

## Getting Your Windows Password

To connect to your Amazon EC2 instance running Windows Server 2008 R2 and IIS Media Services, use the following procedure to retrieve the initial password for the Windows Server Administrator account. You only need to retrieve the password once for your Amazon EC2 instance. When you are finished with this procedure, you'll be able to work with your Amazon EC2 instance as you would any Windows Server computer.

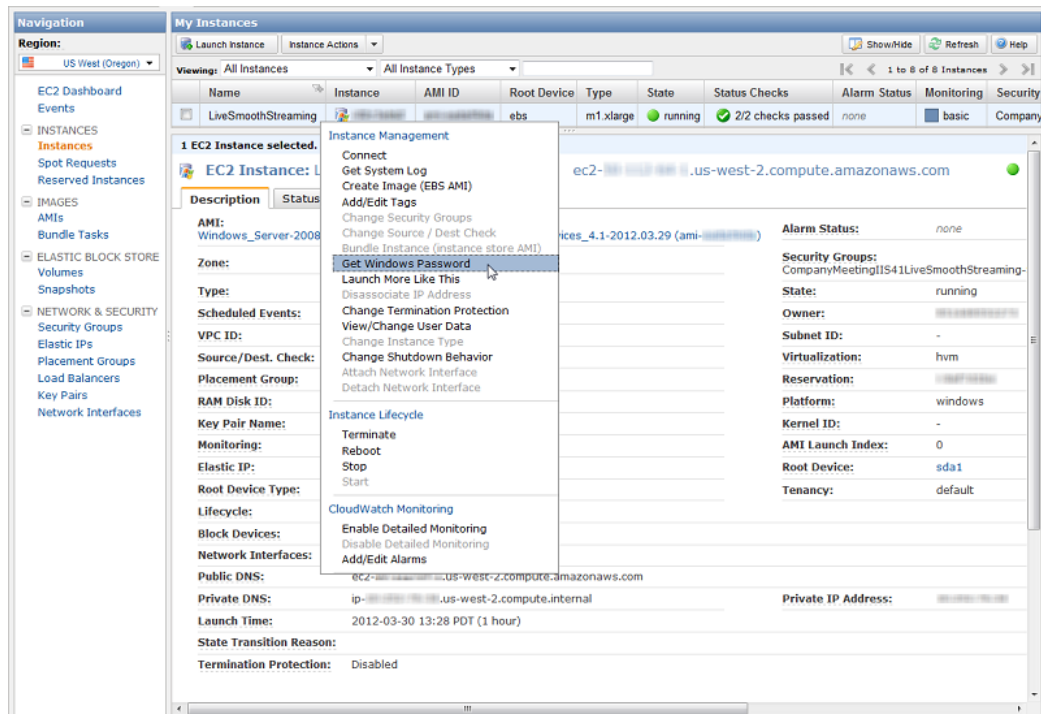
For more information about connecting to an Amazon EC2 instance running Windows, go to [Getting Started Guide AWS Computing Basics for Windows](#).

### Important

Amazon EC2 can take as long as 30 minutes to retrieve your password from Windows Server.

### To get the Windows password for your Amazon EC2 instance

1. Confirm that you can access the Amazon EC2 private key file (the .pem file) that you created in [Creating an Amazon EC2 Key Pair \(p. 175\)](#).
2. Sign in to the AWS Management Console and open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
3. In the **Navigation** pane, in the **Region** list, click the region in which you created the Amazon EC2 instance for Live Smooth Streaming.
4. In the **Navigation** pane, click **Instances**.
5. In the My Instances pane, right-click the instance for which the value of the **Name** column is **LiveSmoothStreaming**, and click **Get Windows Password**.



6. On the Retrieve Default Windows Administrator Password page, click **Browse**, and browse to the location on your computer where you saved the .pem file.
7. Select the .pem file, and the contents of the file appear in the window.



Retrieve Default Windows Administrator Password

Cancel

To access this instance remotely (e.g., Remote Desktop Connection), you will need your Windows Administrator password. A default password was created when the instance was launched and is available encrypted in the system log.

To decrypt your password, you will need your key pair for this instance. Browse to your key pair, or copy & paste the contents of your private key file into the text box below, then click **Decrypt Password**.

Instance: **msk-ec2-ubuntu-ami**

---

\* Required field

**Encrypted Password:**

[REDACTED]

**Key Pair:**

IIS41LiveSmoothStreaming.pem

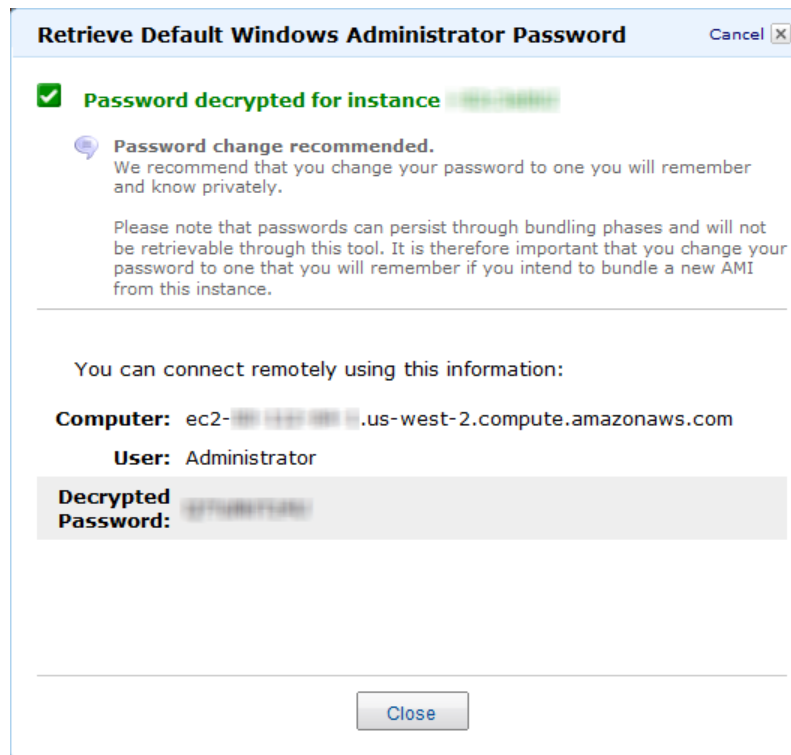
Note: You were prompted to download and save this when you created your key pair.

**Private Key\*:**

-----BEGIN RSA PRIVATE KEY-----  
[REDACTED]  
-----END RSA PRIVATE KEY-----

C:\Users\[REDACTED]\Desktop

- Click **Decrypt Password**.
- Write down the password. You'll need it to connect to the Amazon EC2 instance.



10. *Optional but recommended:* Log into the Windows Server instance that you just launched, and change the password for the default Windows Server account. The username is Administrator.

You may also want to create another user account and add it to the Administrators group. Another administrator account is a safeguard in case you forget your administrator password or have a problem with the Administrator account.

#### Note

For information about how to update the Amazon EC2 Security Group settings for your Windows server so you can access the server using port 3389, see [How can I enable access to the Windows server? \(p. 189\)](#). For information about how to log on to the instance using the Administrator account, see [How can I securely connect to my Amazon EC2 instance running Windows IIS Media Services? \(p. 189\)](#).

Next: [Encoding Your Live Stream \(p. 182\)](#)

## Encoding Your Live Stream

Use the procedure in this section to create a Live Broadcasting Project using Microsoft Expression Encoder 4 Pro SP2 and to publish your live stream to the Live Smooth Streaming publishing point on your Amazon EC2 instance running Windows Server and Windows IIS Media Services.

To learn more about live broadcasting using Microsoft Expression Encoder, go to [Creating a Live Broadcasting Project](#) on the Microsoft Expression website.

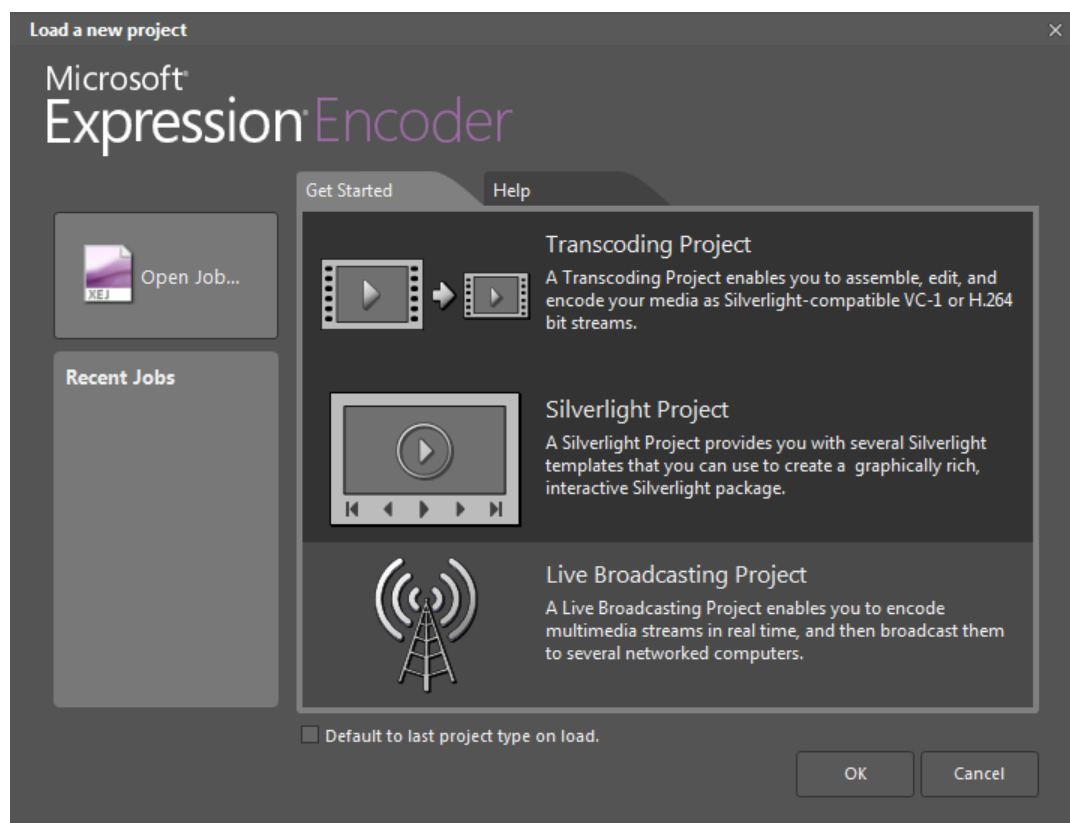
## Note

Microsoft Expression Encoder 4 Pro with Service Pack 2 is not a free download. For more information about features and pricing, go to the [Expression Encoder 4 Pro](#) page on the Microsoft website.

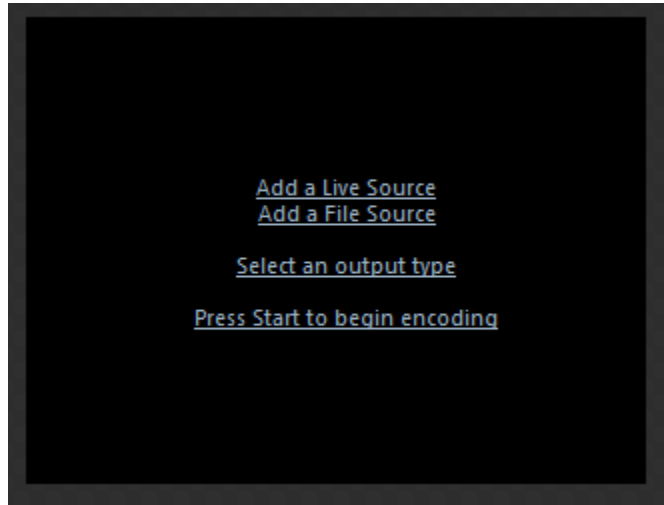
You can also use a third-party encoding tool to encode your video for Live Smooth Streaming. For a list of Microsoft partners that provide encoding software, see the Partners tab on the [IIS Media Services](#) page on the Microsoft website.

## To encode a live broadcast

1. Log into the Amazon EC2 instance that you created for live streaming.
2. On the Windows Start menu, click **All Programs > Microsoft Expression > Microsoft Expression Encoder 4**.
3. In the **Load a new project** dialog box, click **Live Broadcasting Project**.



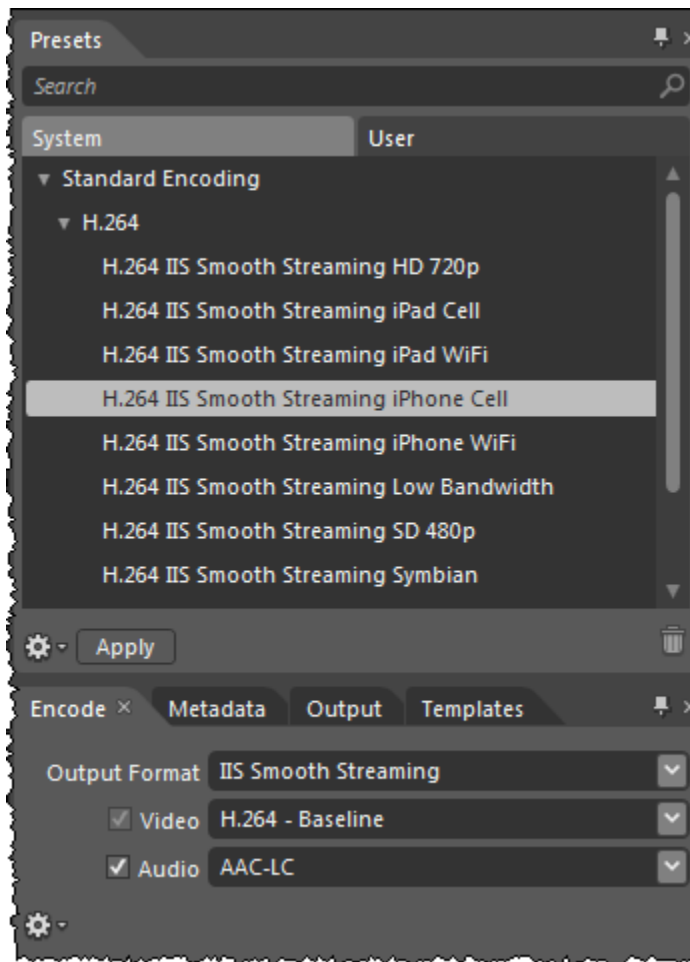
4. Click **Add a Live Source** to use for your live broadcast.



**Note**

You can connect multiple camera devices, such as USB webcams or FireWire (IEEE 1394) digital video cameras. Although you can connect multiple live sources, you can stream only one at a time. For more information about setting up sources for live broadcasts, go to [Set Live Sources](#) on the Microsoft Expression website.

5. On the **Presets** tab, choose the encoding preset that supports the bit rates and encoding requirements for your Live Smooth Streaming scenario. Choose an option that has **IIS Smooth Streaming** in the name.



When you click **Apply**, the **Output Format**, **Video**, and **Audio** settings on the **Encode** tab are automatically updated with the values in the encoding preset that you selected.

For more information about a preset, for example, the number of streams in the output and the codecs used, hover your mouse pointer over a preset name.

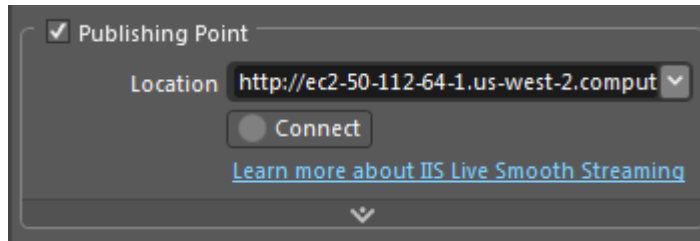
#### Note

Alternatively, you can specify custom settings on the **Encode** tab. For more information, go to the following topics on the Microsoft Expression website:

- [Set Output formats](#)
- [Video settings](#)
- [Audio settings](#)

6. In Microsoft Expression Encoder, click the **Output** tab.
7. On the **Output** tab, check the **Publishing Point** check box.
8. Sign in to the AWS Management Console and open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation/>.
9. In the bottom pane of the AWS CloudFormation console, click the **Outputs** tab.
10. Copy the value of the **LivePublishPointLocation** key, for example, **http://ec2-00-11-22-33.us-west-1.compute.amazonaws.com/LiveSmoothStream.isml**.

11. In Microsoft Expression Encoder, paste the URL that you copied in the previous step into the **Location** field.



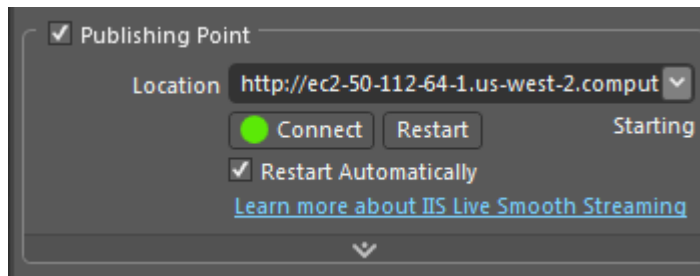
12. Click **Connect** to initiate a connection to the publishing point on your Windows server.
13. When you're prompted for your Publishing Point Administrator Password, enter the following values:
  - **User Name:** Administrator
  - **Password:** the Windows Server password that you retrieved in [Getting Your Windows Password](#) (p. 180).

Then click **OK**.

#### Note

Windows authentication is configured for the default web site on your Windows server so you can connect to the live publishing point on the server from Microsoft Expression Encoder 4 Pro SP2. To learn more about Windows authentication, go to the [IIS website](#). To learn about the authentication mechanisms available in IIS 7, go to the [Microsoft website](#).

14. When a connection is successfully established, the publishing point state changes to **Starting**. In addition, a **Restart** button appears next to the **Connect** button, below the **Location** field in the Publishing Point section.



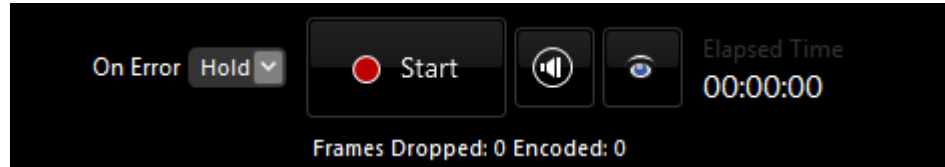
#### Note

The Starting state means that the publishing point is ready to receive live streams. When the live source connects to the publishing point and begins pushing content to it, the Starting state changes to Started, meaning that the publishing point is receiving the live streams.

#### Note

Microsoft Expression Encoder 4 Pro with SP2 utilizes the REST APIs that Windows IIS Media Services 4.1 includes to help you manage live publishing points on the Windows server. For more information, go to the [IIS blog](#).

15. Click **Start** to begin encoding and publishing your live broadcast to the publishing point on your Amazon EC2 instance that is running Windows Server and IIS Media Services.



As the broadcast runs, you can monitor statistics and connections data in the corresponding panels. For more information about how to monitor this data, see the following topics on the Microsoft Expression website:

- [Using the Statistics panel](#)
- [Using the Connections panel](#)

Next: [Viewing Your Live Smooth Stream \(p. 187\)](#)

## Viewing Your Live Smooth Stream

Perform the following procedure to view your live smooth stream using CloudFront. You can also embed the Microsoft Silverlight player code in your own web page.

1. Sign in to the AWS Management Console and open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation/>.
2. Select the stack for live streaming.
3. In the bottom pane of the AWS CloudFormation console, click the **Outputs** tab.
4. Click the value of the **LiveSmoothStreamingPlayer** key, for example, **http://d123.cloudfront.net/LiveSmoothStreamingPlayer.html**.
5. To embed the Silverlight player code into your web page, on the **Outputs** tab, copy the value of the **SilverlightEmbedCode** key.

### Note

Microsoft recommends that viewers have the latest version of Microsoft Silverlight installed for the best playback experience.

6. To view your live stream on an Apple device such as an iPad or an iPhone, display the AWS CloudFormation console from a compatible Apple device, and click the value of the **LiveHLSManifest** key. The manifest URL looks like **http://d123.cloudfront.net/LiveSmoothStream.isml/manifest(format=m3u8-aapl).m3u8**.

For information about where to use the URL to serve various iOS devices, QuickTime, and Safari, go to [HTTP Live Streaming Overview](#) in the iOS Developer Library.

Next: [Deleting Your AWS CloudFormation Live Smooth Streaming Stack \(p. 187\)](#)

## Deleting Your AWS CloudFormation Live Smooth Streaming Stack

When your live event is over, delete the stack that you created for Live Smooth Streaming. This deletes the AWS resources that were created for your live event, and stops the AWS charges for those resources.

### To delete an AWS CloudFormation stack for live streaming

1. Sign in to the AWS Management Console and open the AWS CloudFormation console at <https://console.aws.amazon.com/cloudformation/>.
2. Check the checkbox for the stack, and click **Delete Stack**.
3. Click **Yes, Delete** to confirm.
4. To track the progress of the stack deletion, check the checkbox for the stack, and click the **Events** tab in the bottom frame.

## Frequently Asked Questions

- [What is the price for Live Smooth Streaming using CloudFront? \(p. 188\)](#)
- [Can I deliver my live streaming video to both Smooth Streaming clients and Apple devices? \(p. 188\)](#)
- [How can I set-up a CNAME alias for my Amazon EC2 instance or my CloudFront distribution? \(p. 188\)](#)
- [How can I enable access to the Windows server? \(p. 189\)](#)
- [How can I securely connect to my Amazon EC2 instance running Windows IIS Media Services? \(p. 189\)](#)
- [How can I restrict access to my Live Smooth Streaming content from another domain? \(p. 189\)](#)

## What is the price for Live Smooth Streaming using CloudFront?

To Smooth Stream your live event, you pay only for the AWS resources you consume:

- For pricing information about Amazon EC2 instances running Windows Server, see **Pricing** on the [Amazon EC2 Running Microsoft Windows Server & SQL Server](#) page.
- For pricing information about CloudFront, see [Amazon CloudFront Pricing](#).

There is no charge for using AWS CloudFormation.

## Can I deliver my live streaming video to both Smooth Streaming clients and Apple devices?

Yes. You can use Microsoft Expression Encoder 4 Pro to encode your live video for both Smooth Streaming clients (for example, Microsoft Silverlight) and Apple devices (for example, iPad and iPhone). After your AWS CloudFormation stack is launched, you will find the manifest file URLs both for Live Smooth Streaming (.ism) and for Apple HLS (.m3u8) on the **Outputs** tab of your AWS CloudFormation template.

## How can I set-up a CNAME alias for my Amazon EC2 instance or my CloudFront distribution?

Your Amazon EC2 Windows Server instance comes with an internal and an external DNS name. Amazon EC2 does not provide access to modify these DNS settings. If you want to map an existing domain name to your Amazon EC2 instance running Windows Server, use a DNS service provider such as [Amazon Route 53](#). When using your own domain name, we recommend that you map to the instance's external DNS name using a CNAME, not by using an A record that points to the instance's IP address.

To map your own domain name to your CloudFront distribution, see [Using Alternate Domain Names \(CNAMEs\)](#) (p. 53).



## How can I enable access to the Windows server?

### To enable access to port 3389 on your Windows server via selected IP addresses

By default, the Amazon EC2 security group for your Windows server instance does not have port 3389 enabled; this is the port you use to administer the Windows server. If you want to log on to your Windows server instance, perform the following procedure to enable access via port 3389.

1. Sign in to the AWS Management Console and open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the **Navigation** pane, in the **Region** list, click the Amazon EC2 region in which you used AWS CloudFormation to create your Amazon EC2 instance.
3. In the **Navigation** pane, click **Security Groups**.
4. In the **Security Groups** pane, click the row for which the value of the **Name** column begins with the name of the AWS CloudFormation stack that you created in [Creating an AWS CloudFormation Stack for Live Smooth Streaming](#) (p. 176).
5. In the bottom pane, click the **Inbound** tab.
6. To enable access to your Windows server and specify the client IP addresses that can access the server:
  - a. In the **Create a new rule** list, do not change the default value, **Custom TCP rule**.
  - b. In the **Port range** field, enter **3389**.
  - c. In the **Source** field, enter an IP address or range, or enter the name of another security group. For more information, click **Help**.
  - d. Click **Add Rule**.
  - e. To create additional rules, repeat Steps a through d.
  - f. Click **Apply Rule Changes**.

## How can I securely connect to my Amazon EC2 instance running Windows IIS Media Services?

To connect to your Windows server instance, you must retrieve the initial password for the Administrator account and then use it with Windows Remote Desktop. You'll also need the contents of the private key file that you created, for example, `<keypairname.pem>.pem`. For more information, go to [Getting Started Guide AWS Computing Basics for Windows](#).

## How can I restrict access to my Live Smooth Streaming content from another domain?

Microsoft Silverlight includes support for cross-domain connectivity, which allows the Silverlight player to access content from locations other than the domain where the Smooth Streaming content originates. The security policy system in Silverlight requires that a Silverlight policy file named `ClientAccessPolicy.xml` be downloaded from a target domain before a network connection is allowed to access a network resource under that target domain. A default policy file is already included at the root of the default website on your Windows server running on Amazon EC2. To restrict cross-domain access, log on to your Windows server and update the `ClientAccessPolicy.xml` file.

## Additional Documentation

### Microsoft Documentation

- [IIS Smooth Streaming Deployment Guide](#)
- [IIS Media Services 4.1 Readme](#)
- [IIS Smooth Streaming Management REST Services](#)
- [Configuring Authentication in IIS 7](#)
- [Microsoft Expression Encoder blog](#)
- [Managing live publishing points from Microsoft Expression Encoder 4 Pro SP2](#)
- [Live IIS Smooth Streaming in Expression Encoder 4 Pro](#)
- [Apple HTTP Live Streaming with IIS Media Services](#)

### Amazon Web Services Documentation

- [Amazon EC2 Running Microsoft Windows Server & SQL Server](#)
- [Amazon Elastic Compute Cloud Microsoft Windows Guide](#)
- [Amazon CloudFront](#)
- [AWS CloudFormation](#)

# Restricting Access to Files in a CloudFront Distribution Based on Geographic Location (Geoblocking)

## Topics

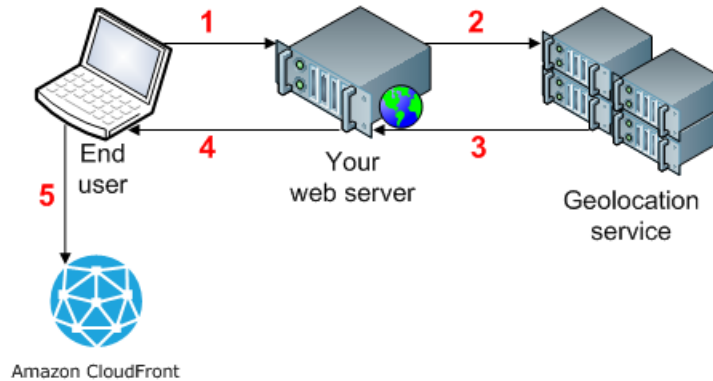
- [Overview of Restricting Access to Files in a CloudFront Distribution Based on Geographic Location \(p. 191\)](#)
- [Creating an Amazon Web Services Account \(p. 193\)](#)
- [Sample Code for Digital Element \(p. 193\)](#)
- [Sample Code for MaxMind \(p. 203\)](#)
- [Frequently Asked Questions \(p. 213\)](#)
- [Additional Services and Documentation \(p. 214\)](#)

## Overview of Restricting Access to Files in a CloudFront Distribution Based on Geographic Location

Amazon CloudFront improves the performance, reliability, and availability of your websites and applications by distributing your web content, such as images, video, and audio to a worldwide network of edge locations. When an end user requests your content, CloudFront serves your content to the user from the edge location that has the lowest latency for that user at that moment. If you have geographic restrictions on where your content can be distributed, you can use CloudFront with a third-party geolocation service to control distribution of your content according to the location of a request. This is known as geoblocking or geotargeting. For example, if a request comes from a country where, for copyright reasons, you are not authorized to distribute your content, you can block the request and direct the requester to a message that explains the situation.

Here's how it works:

1. An end user who is viewing your website requests a web page or a file that is georestricted.
2. Your web application gets the end user's IP address from the request and sends the IP address to a geolocation service. You will need an account with one of these services.
3. The geolocation service determines the geographic location of the end user's IP address and returns the result to your web application.
4. Your web application compares the end user's location with a list of locations where the file can (or can't) be distributed:
  - If the end user is allowed to access the web page or file, your application creates a CloudFront signed URL and returns it to the end user.
  - If the end user is not allowed to access the web page or file, your web application returns the URL of a "you are not authorized" message to the end user.
5. If the end user is allowed to access the web page or file, the end user's browser automatically uses the signed URL to request the file from CloudFront.



Using CloudFront and a third-party geolocation service to restrict access to your content from your application layer gives you full control over your end user's experience. For end users whose access is blocked, your application can display a meaningful message instead of returning an error code. You can also customize the error message you display for your end users according to their location.

The following task list guides you through the process of implementing geoblocking functionality in your applications to restrict access to the content in your CloudFront distribution according to the end user's location.

#### **Task list for restricting access to files in a CloudFront distribution based on geographic location**

1. Get an account with a geolocation service.  
This section provides sample code for Digital Element and for MaxMind, but any geolocation service is supported.
2. If you don't already have an AWS account, create one. For more information, see [Creating an Amazon Web Services Account \(p. 193\)](#)
3. Upload your content to an Amazon Simple Storage Service (S3) bucket. For more information, see the [Amazon S3 documentation](#).
4. Configure Amazon CloudFront and Amazon S3 to serve private content. For more information, see [Private Content Process Overview](#).
5. Write your web application to do the following:
  - a. Send the IP address for each end-user request to the geolocation service.
  - b. Evaluate the return value from the geolocation service (commonly a country code) to determine whether the end user is in a location to which you want CloudFront to distribute your content.
  - c. Either generate a signed URL for your CloudFront content, or block access to the content.

Java, .NET, and PHP sample code is provided below for Digital Element and for MaxMind. See the applicable topic:

- [Sample Code for Digital Element \(p. 193\)](#)
- [Sample Code for MaxMind \(p. 203\)](#)

If you're using another geolocation service, refer to their documentation.

Amazon Web Services provides SDKs for Java, .NET, and PHP. For more information, see the applicable page on the Amazon Web Services website:

- [Java Developer Center](#)
- [Windows & .NET Developer Center](#)
- [PHP Developer Center](#)

## Creating an Amazon Web Services Account

### Note

When you create an account, AWS automatically signs up the account for all services. You are charged only for the services you use.

### To create an AWS account

1. Go to <http://aws.amazon.com>, and click **Create an AWS Account**.
2. Follow the on-screen instructions.  
Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

## Sample Code for Digital Element

The samples in this section show how to get a location from Digital Element from an end user's IP address. The samples also show how to create a signed URL for a requested object if you are allowed to distribute the content to the end user's location.

All sample code was tested before the document was published, but subsequent changes to the Digital Element API could affect whether the samples are still accurate. For the latest information, go to the Digital Element documentation.

See the applicable sample code:

- [Java Sample Code for Digital Element \(p. 193\)](#)
- [.NET Sample Code for Digital Element \(p. 197\)](#)
- [PHP Sample Code for Digital Element \(p. 201\)](#)

### Note

In the code examples, red italicized text is a placeholder. Replace this text with whatever values are appropriate for your situation.

## Java Sample Code for Digital Element

The code example provided here obtains the country code that is associated with an end user's IP address and allows the user to access CloudFront content if the user is in a location where distribution is allowed. For the purposes of the example, the program is authorized to distribute the requested content to any country except Australia (country code AU).

### GetCountryCodeServlet.java

GetCountryCodeServlet.java calls GetDigitalElementCountryCode.java, which is shown later in this article, to ask Digital Element for the country code that is associated with an end user's IP address. If the country

code is not AU (Australia), GetCountryCodeServlet.java calls SignedUrl.java to create a signed URL that the end user can use to access a file in the CloudFront distribution.

```
/*
 * Copyright 2011 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 * http://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 */

// Signed URLs for a private distribution
// Note that Java supports SSL certificates only in DER format,
// so you will need to convert your PEM-formatted file to DER format.
// To do this, you can use openssl:
// openssl pkcs8 -topk8 -nocrypt -in origin.pem -inform PEM -out new.der -outform
// DER
// For the encoder to work correctly, you should also add the
// bouncy castle jar to your project and then add the provider.ds.

import java.io.IOException;
import java.io.PrintWriter;
import java.util.StringTokenizer;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class GetCountryCodeServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    final String GEOAPIURL = "Digital Element URL";
    final String GEOAPITOKEN = "Digital Element user token";
    final String PATHTODER = "path to .der file";
    final String KEYPAIRID = "CloudFront key pair ID";
    final String HTTPORHTTPS = "https";
    final String CFDISTRIBUTION = "dxxxx.cloudfront.net";
    final String CFPATH = "CloudFront URL for file";
    // date and time that CloudFront's signed URL expires,
    // in Coordinated Universal Time
    final String EXPIRETS = "2012-11-14T22:20:00.000Z";
    final String BLOCKEDCOUNTRY="AU";

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {

        String ip = null;
        StringTokenizer st = null;
        PrintWriter out = response.getWriter();
```

```
String headers = request.getHeader("X-FORWARDED-FOR");

if (headers!= null){
    st = new StringTokenizer(headers,",");

    while (st.hasMoreTokens()) {
        ip = st.nextToken();
    }
}

//Get the client's IP addr in case X-Forwarded-IP header doesn't exist

if (ip == null) ip = request.getRemoteAddr();

try {
    GetDigitalElementCountryCode country = new GetDigitalElementCountryCode(
GEOAPIURL,GEOAPITOKEN );

    if ( !country.getCountry(ip).equalsIgnoreCase(BLOCKEDCOUNTRY)){

        SignedUrl myApp = new SignedUrl(KEYPAIRID,PATHTODER);
        out.println(myApp.getSignedHash(HTTPORHTTPS,CFDISTRIBUTION,CFPATH,EX
PIRETS));

    }else {
        out.println("You cannot access this link.");
    }
} catch (Exception e1) {
    e1.printStackTrace();
}
}
```

## GetDigitalElementCountryCode.java

GetDigitalElementCountryCode.java sends Digital Element a request that includes an end user's IP address. The return value is a country code.

```
/*
 * Copyright 2011 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 * http://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 */

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

import org.w3c.dom.Document;
```

```
import org.w3c.dom.Element;
import org.w3c.dom.NodeList;

public class GetDigitalElementCountryCode {

    private static String geoApiEndPoint;
    private static String apiToken;

    GetDigitalElementCountryCode(String mygeoApiEndPoint, String myapiToken){
        geoApiEndPoint = mygeoApiEndPoint;
        apiToken = myapiToken;
    }

    public String getCountry(String enduserIP) throws Exception {

        String geoApiURL = "http://" + geoApiEndPoint + "?u=" + apiToken + "&ip=" + end
        userIP;

        DocumentBuilderFactory docBuilderFactory = DocumentBuilderFactory.newInstance();
        DocumentBuilder docBuilder = docBuilderFactory.newDocumentBuilder();
        Document doc = docBuilder.parse(geoApiURL);
        // normalize text representation
        doc.getDocumentElement().normalize();

        NodeList listPersons = doc.getElementsByTagName("response");
        Element el = (Element)listPersons.item(0);
        String country = el.getAttribute("edge-two-letter-country");

        return country;
    }
}
```

## SignedUrl.java

SignedUrl.java creates a signed URL that the end user can use to access a file in the CloudFront distribution.

```
/*
 * Copyright 2011 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 * http://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 */

import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.security.Security;
```



```
import java.text.ParseException;

import org.jets3t.service.CloudFrontService;
import org.jets3t.service.CloudFrontServiceException;
import org.jets3t.service.utils.ServiceUtils;

public class SignedUrl {
    // Signed URLs for a private distribution
    // Note that Java supports SSL certificates only in DER format,
    // so you need to convert your PEM-formatted file to DER format.
    // To do this, you can use openssl:
    // openssl pkcs8 -topk8 -nocrypt -in origin.pem -inform PEM -out new.der -
    outform DER
    // For the encoder to work correctly, you should also add the
    // bouncy castle jar to your project and then add the provider.ds.

    private static String keyPairId;
    private static String privateKeyFilePath;

    SignedUrl(String mykeyPairId, String myprivateKeyFilePath){
        keyPairId = mykeyPairId;
        privateKeyFilePath = myprivateKeyFilePath;
    }

    public String getSignedHash(String protocol, String cfDistribution, String
    objectUri, String expTime) throws FileNotFoundException, IOException,
    CloudFrontServiceException, ParseException{

        Security.addProvider(new org.bouncycastle.jce.provider.BouncyCastlePro
        vider());

        // Convert your DER file into a byte array.

        byte[] derPrivateKey = ServiceUtils.readInputStreamToBytes(new FileInput
        Stream(privateKeyFilePath));

        // Generate a "canned" signed URL to allow access to a
        // specific distribution and object

        String signedUrlCanned = CloudFrontService.signUrlCanned(
            protocol+ "://" + cfDistribution + "/" + objectUri, // Resource URL or
            Path
            keyPairId, // Certificate identifier,
            // an active trusted signer for the distribution
            derPrivateKey, // DER Private key data
            ServiceUtils.parseIso8601Date(expTime) // DateLessThan
            );

        return signedUrlCanned;
    }
}
```

## .NET Sample Code for Digital Element

The following sample application gets the IP address of the end user and sends the IP address to Digital Element. Digital Element returns the country code (in XML format) that corresponds with the end user's IP address. The application parses the XML and evaluates whether the value returned by Digital Element

matches the blocked country code. If the end user's country is blocked, the application displays a message to that effect. If the end user's country is not blocked, the application creates a signed URL that expires in one minute, performs the substitutions necessary to ensure that the URL doesn't include any invalid characters, and redirects the user's browser to the signed URL.

```
<%@ Page Language="C#" AutoEventWireup="true" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head id="Head1" runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <%=GetContent()%>
        </div>
    </form>
</body>
</html>

<%@ Import Namespace="System.Linq" %>
<%@ Import Namespace="System.Xml.Linq" %>
<%@ Import Namespace="System.Security.Cryptography" %>
<%@ Import Namespace="System.Net" %>
<%@ Import Namespace="System.IO" %>

<script runat="server">

    // Key pair ID for the CloudFront key pair
    private const string KEYPAIR_ID = "CloudFront key pair ID";

    // Private key for the CloudFront key pair.
    // The value is derived from opensslkey.
    private const string PRIVATE_KEY = "private key";

    // JSON policy statement used in the expiring URL
    private const string POLICY = "{ \"Statement\": [ { \"Resource\": \"{0}\", \"Condition\": { \"DateLessThan\": { \"AWS:EpochTime\": {1} } } } ] }";

    // Digital Element user token to be passed to geolocation service call

    private const string USERTOKEN = "Digital Element user token";
    private const string GEOAPIURL = "Digital Element URL";

    // GEO IP service URL with parameters:
    // {0} = User Token and {1} = IP Address
    private const string SERVICEURL = GEOAPIURL + "?u={0}&ip={1}";

    // Array of countries to block
    private static readonly string[] COUNTRIES_TO_BLOCK = new String[] { "US" };

    private const string BLOCKED_MSG = "Your access to this content is blocked because you're visiting from '{0}'.";

    /// <summary>
```

```
/// Returns the IP address coming from the request object.
/// </summary>
/// <returns>The IP address for the request.</returns>
private string GetOriginIpAddress()
{
    // .NET provides Request.UserHostAddress to get the
    // remote IP address, but this could be the IP address of the
    // last proxy in a chain, for example, an Elastic Load Balancer.
    // Instead, use the HTTP_X_FORWARDED_FOR header if one exists.
    string forwardedIpAddresses = this.Request.ServerVariables["HTTP_X_FORWARDED_FOR"];

    if (string.IsNullOrEmpty(forwardedIpAddresses))
    {
        // Simply return the UserHostAddress.
        return Request.UserHostAddress;
    }
    else
    {
        // Get the last item in the list.
        return forwardedIpAddresses.Split(',').Last().Trim();
    }
}

/// <summary>
/// This function returns the country code
/// associated with the IP address in the request object.
/// </summary>
/// <returns>The country code for the request.</returns>
private string GetCountryCodeFromIP()
{
    var ipAddress = GetOriginIpAddress();
    var serviceURL = String.Format(SERVICEURL, Server.UrlEncode(USERTOKEN),
Server.UrlEncode(ipAddress));

    try
    {
        var xDoc = XDocument.Load(serviceURL);
        var res = (from w in xDoc.Descendants("response") select w).First();

        return res.Attribute("edge-two-letter-country").Value.ToUpper();
    }
    catch(Exception ex)
    {
        // There was an error in making the web request.
        this.Response.Write(serviceURL + "<br><br>");
        this.Response.Write(ex.Message);
        this.Response.End();
    }
    return null;
}

/// <summary>
/// This function returns a signed URL that will expire in 1 minute.
/// For more information, see "Create a URL Signature Using C# and the
/// .NET Framework" in the Amazon CloudFront Developer Guide:
/// http://docs.amazonwebservices.com/AmazonCloudFront/latest/DeveloperGuide/CreateSignatureInCSharp.html?r=4472

```

```
/// </summary>
/// <param name="resourceUrl"></param>
/// <returns></returns>
private string GetSignedURL(string resourceUrl)
{
    // Compute expiration date.
    var endTimeSpanFromNow = new TimeSpan(0, 1, 0);
    var intervalEnd = (DateTime.UtcNow.Add(endTimeSpanFromNow)) - new Date
Time(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);
    var endTimestamp = (int)intervalEnd.TotalSeconds; // Timestamp must be a
whole number
    var expires = endTimestamp.ToString();
    var strPolicy = string.Format(POLICY, resourceUrl, expires);

    // Encrypt the policy.
    var bufferPolicy = Encoding.ASCII.GetBytes(strPolicy);
    var cryptoSHA1 = new SHA1CryptoServiceProvider();
    bufferPolicy = cryptoSHA1.ComputeHash(bufferPolicy);
    var providerRSA = new RSACryptoServiceProvider();
    providerRSA.FromXmlString(PRIVATE_KEY);
    var rsaFormatter = new RSAPKCS1SignatureFormatter(providerRSA);
    rsaFormatter.SetHashAlgorithm("SHA1");
    var signedPolicyHash = rsaFormatter.CreateSignature(bufferPolicy);
    var strSignedPolicy = System.Convert.ToBase64String(signedPolicyHash);

    // Build the query string with the expiration, policy signature,
    // and CloudFront key pair ID.
    var queryString = "Expires={0}&Signature={1}&Key-Pair-Id={2}";
    queryString = String.Format(queryString, Server.UrlEncode(expires),
Server.UrlEncode(strSignedPolicy), Server.UrlEncode(KEYPAIR_ID));
    var urlString = resourceUrl + "?" + queryString;
    return urlString;
}

/// <summary>
/// Return a message saying this is blocked because of your country, or
/// return an image tag.
/// </summary>
/// <returns></returns>
public string GetContent()
{
    var country = GetCountryCodeFromIP();
    if (COUNTRIES_TO_BLOCK.Contains(country))
    {
        // The country returned from the call to the geolocation service
        // is listed in the array of blocked countries.
        return string.Format(BLOCKED_MSG, country);
    }
    else
    {
        // The country returned from the call to the geolocation service
        // is NOT listed in the array of blocked countries
        // Get a CloudFront signed URL for the content and display it.
        var url = GetSignedURL("CloudFront URL");
        var img = "<img src='{0}' />";
        return String.Format(img, url);
    }
}
```

```
}  
</script>
```

## PHP Sample Code for Digital Element

The following sample application gets the IP address of the end user and sends the IP address to Digital Element. Digital Element returns the country code (in XML format) that corresponds to the end user's IP address. The application then parses the XML, displays the country code that is blocked, and evaluates whether the value returned by Digital Element matches the blocked country code. If the end user's country is not blocked, the application displays a "You are not blocked" message, uses a canned policy to create a signed URL that expires in five minutes, performs the substitutions necessary to ensure that the URL doesn't include any invalid characters, and redirects the user's browser to the signed URL. If the end user's country is blocked, the application displays a "You are blocked" message and a graphic.

```
<!DOCTYPE html>  
<html>  
<head>  
    <title>Geoblocking Test</title>  
</head>  
<body>  
    <h1>Geoblocking Test</h1>  
  
<?php  
// Configure the private key (make sure this information is secure).  
$private_key_filename = 'path to private key';  
$key_pair_id          = 'CloudFront key pair ID';  
  
/*  
 * Configure the geoblocking parameters. The following variables  
 * describe the two-letter country to be blocked, the  
 * CloudFront URL for the file that you want to secure,  
 * and the expiry time of the URL. Change these values as needed.  
 */  
$blocked_geo = 'uk';  
$asset_path  = 'CloudFront URL for the object';  
$expires     = time() + 300; // (5 minutes from now)  
  
// Configure the URL to the geoblocking service.  
$token       = 'Digital Element user token';  
$address     = 'Digital Element URL';  
$remote_ip   = get_remote_ip_address();  
$service_url = $address . '?u=' . $token . '&ip=' . $remote_ip;  
  
// Call the web service using the configured URL.  
$ch = curl_init();  
curl_setopt($ch, CURLOPT_URL, $service_url);  
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);  
$ws_response = curl_exec($ch);  
  
// Parse the response with SimpleXML and get the geoblocking value.  
$xml        = new SimpleXMLElement($ws_response);  
$edge_geo   = $xml->response->attributes()->{'edge-two-letter-country'};  
  
echo '<p>The country being blocked is: ' . strtoupper($blocked_geo) . '</p>';  
  
if ($edge_geo != $blocked_geo)
```

```
{
    echo '<p>Your country is: ' . strtoupper($edge_geo) . '</p>';
    echo '<p>You are not blocked.</p>';
    $signed_url = create_signed_url($asset_path, $private_key_filename,
    $key_pair_id, $expires);
    echo 'Your country is: ' . strtoupper($edge_geo) . '</p>';
    echo '<p>You are blocked.</p>';
    $blocked_url = 'http://s3.amazonaws.com/<Amazon S3 bucket>/blocked-image.jpg';

    echo '

</body>
</html>
```

## Sample Code for MaxMind

The samples in this section show how to get a location from MaxMind from an end user's IP address and, if you are authorized to distribute the requested object to the user's location, how to create a signed URL for the object.

All sample code was tested before the document was published, but subsequent changes to the MaxMind API could affect whether the samples are still accurate. For the latest information, go to the MaxMind documentation.

See the applicable sample code:

- [Java Sample Code for MaxMind \(p. 204\)](#)
- [.NET Sample Code for MaxMind \(p. 210\)](#)
- [PHP Sample Code for MaxMind \(p. 208\)](#)

## Java Sample Code for MaxMind

### GetCountryCodeServlet.java

GetCountryCodeServlet.java calls GetMaxMindCountryCode.java, which is shown later in this article, to ask MaxMind for the country code that is associated with an end user's IP address. If the country code is not AU (Australia), GetCountryCodeServlet.java calls SignedUrl.java to create a signed URL that the end user can use to access a file in the CloudFront distribution.

```
/*
 * Copyright 2011 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 * http://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 */

// Signed URLs for a private distribution
// Note that Java supports SSL certificates only in DER format,
// so you will need to convert your PEM-formatted file to DER format.
// To do this, you can use openssl:
// openssl pkcs8 -topk8 -nocrypt -in origin.pem -inform PEM -out new.der -outform
// DER
// For the encoder to work correctly, you should also add the
// bouncy castle jar to your project and then add the provider.ds.

import java.io.IOException;
import java.io.PrintWriter;
import java.util.StringTokenizer;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class GetCountryCodeServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    final String GEOAPIURL = "MaxMind URL";
    final String GEOAPITOKEN = "MaxMind user token";
    final String PATHTODER = "path to .der file";
    final String KEYPAIRID = "CloudFront key pair ID";
    final String HTTPORHTTPS = "https";
    final String CFDISTRIBUTION = "dxxxxx.cloudfront.net";
    final String CFPATH = "CloudFront URL for file";
    // date and time that CloudFront's signed URL expires,
    // in Coordinated Universal Time
    final String EXPIRETS = "2012-11-14T22:20:00.000Z";
    final String BLOCKEDCOUNTRY="AU";
```



```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

    String ip = null;
    StringTokenizer st = null;
    PrintWriter out = response.getWriter();

    String headers = request.getHeader("X-FORWARDED-FOR");

    if (headers!= null){
        st = new StringTokenizer(headers,",");

        while (st.hasMoreTokens()) {
            ip = st.nextToken();
        }
    }

    //Get the client's IP addr in case X-Forwarded-IP header doesn't exist.

    if (ip == null) ip = request.getRemoteAddr();

    try {

        GetMaxMindCountryCode country = new GetMaxMindCountryCode("GEOAPI
URL","GEOAPITOKEN");

        if ( !country.getCountry(ip).equals(BLOCKEDCOUNTRY)){

            SignedUrl myApp = new SignedUrl(KEYPAIRID,PATHTODER);
            out.println(myApp.getSignedHash(HTTPORHTTPS,CFDISTRIBUTION,CFPATH,EX
PIRETS));

        }else {
            out.println("You cannot access this link.");
        }
    } catch (Exception e1) {
        e1.printStackTrace();
    }
}
```

## GetMaxMindCountryCode.java

GetMaxMindCountryCode.java sends MaxMind a request that includes an end user's IP address. The return value is a country code.

```
/*
 * Copyright 2011 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 * http://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
```

```
* express or implied. See the License for the specific language governing
* permissions and limitations under the License.
*/

import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.URL;
import java.net.URLConnection;

public class GetMaxMindCountryCode {

    private static String geoApiEndPoint;
    private static String apiToken;

    GetMaxMindCountryCode(String mygeoApiEndPoint, String myapiToken){
        geoApiEndPoint = mygeoApiEndPoint;
        apiToken = myapiToken;
    }

    public String getCountry(String enduserIP) throws Exception {
        String geoApiURL = "http://" + geoApiEndPoint + "?l=" + apiToken + "&i=" + enduserIP;

        // Call to MaxMind API.
        URL url = new URL(geoApiURL);
        URLConnection urlConn = url.openConnection();

        urlConn.setUseCaches(false);

        InputStreamReader in = new InputStreamReader((InputStream) urlConn.getCon
tent());
        BufferedReader buff = new BufferedReader(in);

        return buff.readLine();
    }
}
```

## SignedUrl.java

SignedUrl.java creates a signed URL that the end user can use to access a file in the CloudFront distribution.

```
/*
 * Copyright 2011 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 * http://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 */
```

```
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.security.Security;
import java.text.ParseException;

import org.jets3t.service.CloudFrontService;
import org.jets3t.service.CloudFrontServiceException;
import org.jets3t.service.utils.ServiceUtils;

public class SignedUrl {
    // Signed URLs for a private distribution
    // Note that Java supports SSL certificates only in DER format,
    // so you need to convert your PEM-formatted file to DER format.
    // To do this, you can use openssl:
    // openssl pkcs8 -topk8 -nocrypt -in origin.pem -inform PEM -out new.der -
    outform DER
    // For the encoder to work correctly, you should also add the
    // bouncy castle jar to your project and then add the provider.ds.

    private static String keyPairId;
    private static String privateKeyFilePath;

    SignedUrl(String mykeyPairId, String myprivateKeyFilePath){
        keyPairId = mykeyPairId;
        privateKeyFilePath = myprivateKeyFilePath;
    }

    public String getSignedHash(String protocol, String cfDistribution, String
    objectUri, String expTime) throws FileNotFoundException, IOException,
    CloudFrontServiceException, ParseException{

        Security.addProvider(new org.bouncycastle.jce.provider.BouncyCastlePro
        vider());

        // Convert your DER file into a byte array.

        byte[] derPrivateKey = ServiceUtils.readInputStreamToBytes(new FileInput
        Stream(privateKeyFilePath));

        // Generate a "canned" signed URL to allow access to a
        // specific distribution and object.

        String signedUrlCanned = CloudFrontService.signUrlCanned(
            protocol+ "://" + cfDistribution + "/" + objectUri, // resource URL or
path
            keyPairId, // Certificate identifier,
            // an active trusted signer for the distribution
            derPrivateKey, // DER private key data
            ServiceUtils.parseIso8601Date(expTime) // DateLessThan
        );

        return signedUrlCanned;
    }
}
```

## PHP Sample Code for MaxMind

The following sample application gets the IP address of the end user and sends the IP address to MaxMind. MaxMind returns the country code that corresponds to the end user's IP address. The application then displays the country code that is blocked and evaluates whether the value returned by MaxMind matches the blocked country code. If the end user's country is not blocked, the application displays a "You are not blocked" message, uses a canned policy to create a signed URL that expires in five minutes, performs the substitutions necessary to ensure that the URL doesn't include any invalid characters, and redirects the user's browser to the signed URL. If the end user's country is blocked, the application displays a "You are blocked" message and a graphic.

```
<!DOCTYPE html>
<html>
<head>
    <title>Geoblocking Test</title>
</head>
<body>
    <h1>Geoblocking Test</h1>

<?php
// Configure the private key (make sure this information is secure).
$private_key_filename = 'path to private key';
$key_pair_id          = 'CloudFront key pair ID';

/*
 * Configure the geoblocking parameters. The following variables
 * describe the two-letter country to be blocked, the
 * CloudFront URL for the file that you want to secure,
 * and the expiry time of the URL. Change these values as needed.
 */
$blocked_geo = 'gb';
$asset_path  = 'CloudFront URL for the object';
$expires     = time() + 300; // (5 minutes from now)

// Configure the URL to the geolocation service.
$token       = 'MaxMind user token';
$address      = 'MaxMind URL';
$remote_ip   = get_remote_ip_address();
$service_url = $address . '?l=' . $token . '&i=' . $remote_ip;

// Call the web service using the configured URL.
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $service_url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
$ws_response = curl_exec($ch);

$edge_geo = $ws_response;

echo '<p>The country being blocked is: ' . strtoupper($blocked_geo) . '</p>';

if ($edge_geo != strtoupper($blocked_geo))
{
    echo '<p>Your country is: ' . strtoupper($edge_geo) . '</p>';
    echo '<p>You are not blocked.</p>';
    $signed_url = create_signed_url($asset_path, $private_key_filename,
    $key_pair_id, $expires);
    echo '';
}
```

```
}
else
{
    echo '<p>Your country is: ' . strtoupper($edge_geo) . '</p>';
    echo '<p>You are blocked.</p>';
    $blocked_url = 'http://s3.amazonaws.com/<Amazon S3 bucket>/blocked-image.jpg';

    echo '';
}

// Function definitions

function get_remote_ip_address()
{
    // Check to see if an HTTP_X_FORWARDED_FOR header is present.

    if($_SERVER['HTTP_X_FORWARDED_FOR'])
    {
        // If the header is present, use the last IP address.
        $temp_array = explode(',', $_SERVER['HTTP_X_FORWARDED_FOR']);
        $temp_ip_address = $temp_array[count($temp_array) - 1];
    }
    else
    {
        // If the header is not present, use the
        // default server variable for remote address.
        $temp_ip_address = $_SERVER['REMOTE_ADDR'];
    }

    return $temp_ip_address;
}

function create_signed_url($asset_path, $private_key_filename, $key_pair_id,
$expires)
{
    // Build the policy.
    $scanned_policy = '{"Statement":[{"Resource":"' . $asset_path
        . '","Condition":{"DateLessThan":{"AWS:EpochTime":"' . $expires . '}}}]}';

    // Sign the policy.
    $signature = rsa_shal_sign($scanned_policy, $private_key_filename);

    // Make the signature contains only characters that
    // can be included in a URL.
    $encoded_signature = url_safe_base64_encode($signature);

    // Combine the above into a properly formed URL name
    $temp_signed_url = $asset_path . '?Expires=' . $expires . '&Signature='
        . $encoded_signature . '&Key-Pair-Id=' . $key_pair_id;

    return $temp_signed_url;
}
```

```
function rsa_shal_sign($policy, $private_key_filename)
{
    $signature = '';

    // Load the private key.
    $fp = fopen($private_key_filename, 'r');
    $private_key = fread($fp, 8192);
    fclose($fp);

    $private_key_id = openssl_get_privatekey($private_key);

    // Compute the signature.
    openssl_sign($policy, $signature, $private_key_id);

    // Free the key from memory.
    openssl_free_key($private_key_id);

    return $signature;
}

function url_safe_base64_encode($value)
{
    $encoded = base64_encode($value);

    // Replace characters that cannot be included in a URL.
    return str_replace(array('+', '=', '/'), array('-', '_', '~'), $encoded);
}
?>

</body>
</html>
```

## .NET Sample Code for MaxMind

The following sample application gets the IP address of the end user and sends the IP address to MaxMind. MaxMind returns the country code that corresponds with the end user's IP address. The application then evaluates whether the value returned by Digital Element matches the blocked country code. If the end user's country is blocked, the application displays a message to that effect. If the end user's country is not blocked, the application creates a signed URL that expires in one minute, performs the substitutions necessary to ensure that the URL doesn't include any invalid characters, and redirects the user's browser to the signed URL.

```
<%@ Page Language="C#" AutoEventWireup="true" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "ht
tp://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head id="Head1" runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <%=GetContent()%>
        </div>
```

```
</form>
</body>
</html>

<%@ Import Namespace="System.Linq" %>
<%@ Import Namespace="System.Xml.Linq" %>
<%@ Import Namespace="System.Security.Cryptography" %>
<%@ Import Namespace="System.Net" %>
<%@ Import Namespace="System.IO" %>

<script runat="server">

    // Key pair ID for the CloudFront key pair
    private const string KEYPAIR_ID = "CloudFront key pair ID";

    // Private key for the CloudFront key pair.
    // The value is derived from opensslkey.
    private const string PRIVATE_KEY = "private key";

    // JSON policy statement used in the expiring URL
    private const string POLICY = "{{\"Statement\": [{\"Resource\": \"{0}\", \"Con-
dition\": {\"DateLessThan\": {\"AWS:EpochTime\": {1}}}}]}}";

    // User token to be passed in to GEO IP service call
    private const string USERTOKEN = "user token";

    // Geolocation service URL with parameters:
    // {0} = User Token and {1} = IP address
    private const string SERVICEURL = "http://geoip3.maxmind.com/a?l={0}&i={1}";

    // Array of countries to block
    private static readonly string[] COUNTRIES_TO_BLOCK = new String[] { "US" };

    private const string BLOCKED_MSG = "Your access to this content is blocked
because you're visiting from '{0}'.";

    /// <summary>
    /// Returns the IP address coming from the request object.
    /// </summary>
    /// <returns>The IP address for the request.</returns>
    private string GetOriginIpAddress()
    {
        // .NET provides Request.UserHostAddress to get the
        // remote IP address, but this could be the IP address of the
        // last proxy in a chain, for example, an Elastic Load Balancer.
        // Instead use the HTTP_X_FORWARDED_FOR header if one exists.
        string forwardedIpAddresses = this.Request.ServerVariables["HTTP_X_FORWAR
DED_FOR"];

        if (string.IsNullOrEmpty(forwardedIpAddresses))
        {
            // Return the UserHostAddress.
            return Request.UserHostAddress;
        }
        else
        {
            // Get the last item in the list.
```

```

        return forwardedIpAddresses.Split(',').Last().Trim();
    }
}

/// <summary>
/// This function returns the country code
/// associated with the IP address in the request object.
/// </summary>
/// <returns>The country code for the request.</returns>
private string GetCountryCodeFromIP()
{
    var ipAddress = GetOriginIpAddress();
    var serviceURL = String.Format(SERVICEURL, Server.UrlEncode(USERTOKEN),
Server.UrlEncode(ipAddress));

    try
    {
        var webReq = HttpWebRequest.Create(serviceURL);
        var webRes = webReq.GetResponse().GetResponseStream();
        var sr = new StreamReader(webRes);
        var strRes = sr.ReadToEnd();
        sr.Close();
        return strRes.Trim().ToUpper();
    }
    catch(Exception ex)
    {
        // There was an error in making the web request.
        this.Response.Write(serviceURL + " <br><br>");
        this.Response.Write(ex.Message);
        this.Response.End();
    }
    return null;
}

/// <summary>
/// This function returns a signed URL that will expire
/// in 1 minute. For more information, see "Create a URL Signature
/// Using C# and the .NET Framework" in the Amazon CloudFront Developer
Guide:
/// http://docs.amazonwebservices.com/AmazonCloudFront/latest/Developer
Guide/CreateSignatureInCSharp.html?r=4472
/// </summary>
/// <param name="resourceUrl"></param>
/// <returns></returns>
private string GetSignedURL(string resourceUrl)
{
    // Compute expiration date.
    var endTimeSpanFromNow = new TimeSpan(0, 1, 0);
    var intervalEnd = (DateTime.UtcNow.Add(endTimeSpanFromNow)) - new Date
Time(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);
    var endTimestamp = (int)intervalEnd.TotalSeconds; // Timestamp must be a
whole number
    var expires = endTimestamp.ToString();
    var strPolicy = string.Format(POLICY, resourceUrl, expires);

    // Encrypt the policy.
    var bufferPolicy = Encoding.ASCII.GetBytes(strPolicy);
    var cryptoSHA1 = new SHA1CryptoServiceProvider();

```



```
bufferPolicy = cryptoSHA1.ComputeHash(bufferPolicy);
var providerRSA = new RSACryptoServiceProvider();
providerRSA.FromXmlString(PRIVATE_KEY);
var rsaFormatter = new RSAPKCS1SignatureFormatter(providerRSA);
rsaFormatter.SetHashAlgorithm("SHA1");
var signedPolicyHash = rsaFormatter.CreateSignature(bufferPolicy);
var strSignedPolicy = System.Convert.ToBase64String(signedPolicyHash);

// Build the query string with the expiration, policy signature,
// and CloudFront key pair ID.
var queryString = "Expires={0}&Signature={1}&Key-Pair-Id={2}";
queryString = String.Format(queryString, Server.UrlEncode(expires),
Server.UrlEncode(strSignedPolicy), Server.UrlEncode(KEYPAIR_ID));
var urlString = resourceUrl + "?" + queryString;
return urlString;
}

/// <summary>
/// Return a message saying this is blocked because of your location,
/// or return an image tag.
/// </summary>
/// <returns></returns>
public string GetContent()
{
    var country = GetCountryCodeFromIP();
    if (COUNTRIES_TO_BLOCK.Contains(country))
    {
        // The country returned from the call to the geolocation service
        // is listed in the array of blocked countries.
        return string.Format(BLOCKED_MSG, country);
    }
    else
    {
        // The country returned from the call to the geolocation service
        // is NOT listed in the array of blocked countries
        // Get a signed URL for the content and display it.
        var url = GetSignedURL("CloudFront URL");
        var img = "<img src='{0}' />";
        return String.Format(img, url);
    }
}
</script>
```

## Frequently Asked Questions

### How can I ensure that I retrieve the correct IP address of the end user visiting my website?

You can use a variety of methods to get the IP address of the end user visiting your website. Here are two possible methods:

- If your web server is not connected to the Internet through a load balancer, you can use a web server variable to get the remote IP address. However, this IP address isn't always the end user's IP address—it can also be the IP address of a proxy server, depending on how the end user is connected to the Internet.
- If your web server is connected to the Internet through a load balancer, a web server variable may contain the IP address of the load balancer, not the IP address of the end user. In this configuration, we recommend that you use the last IP address in the X-Forwarded-For http header. This header

typically contains more than one IP address, most of which are for proxies or load-balancers. The last IP address in the list is the one most likely to be associated with the end user's geographic location.

If your web server is not connected to a load balancer, we recommend that you use web server variables instead of the `X-Forwarded-For` header in order to avoid IP address spoofing. The sample code in this document uses the `X-Forwarded-For` header if the header is present. If you do not want to use this method to get the IP address of the end user, you can edit the sample code.

#### **Can I use any third-party geolocation service to restrict access to my content in CloudFront?**

Yes. You will need an account with the third-party service to call their API, and you will need to modify the sample code accordingly.

#### **What is the cost of using this solution?**

The cost of using a third-party geolocation service will depend on which service provider you use. Current pricing for CloudFront usage is available on the [Amazon CloudFront Pricing](#) page. There are no additional CloudFront charges for using the CloudFront private-content feature.

#### **Can I use location information other than country to block access to my content?**

If your geolocation service provides information in addition to the country code, your application can use that information to determine whether you can distribute your content to the end user. Then your application can generate a CloudFront signed URL as described in this tutorial or in [Using a Signed URL to Serve Private Content](#) in the [Amazon CloudFront Developer Guide](#).

#### **What should I do if the third-party service is not returning the correct information about an end user?**

Confirm that you are correctly calling the API provided by the third-party geolocation service, and that you are using the correct IP address for the end user. If you are still encountering issues with the third-party service or with the accuracy of the data that you receive from the service, contact the service vendor directly.

## **Additional Services and Documentation**

### **Digital Element Services and Documentation**

For information about Digital Element services, see the [Digital Element website](#).

Documentation for Digital Element services is available only with a Digital Element account.

### **MaxMind Services and Documentation**

MaxMind offers a variety of geolocation services and other web services, including the following services:

- MaxMind GeoIP Omni Web Service, [http://www.maxmind.com/app/web\\_services\\_omni](http://www.maxmind.com/app/web_services_omni)
- MaxMind JavaScript Web Service, <http://www.maxmind.com/app/javascript>
- Other MaxMind web services, [http://www.maxmind.com/app/web\\_services](http://www.maxmind.com/app/web_services)

The download distribution for each MaxMind API includes documentation and sample programs.

For more information, see [Using MaxMind with Amazon CloudFront](#).

## Amazon Web Services Documentation

- CloudFront, <http://aws.amazon.com/documentation/cloudfront>
- Amazon S3, <http://aws.amazon.com/documentation/s3/>

# Amazon CloudFront Resources

---

The following table lists related resources that you'll find useful as you work with this service.

Resource	Description
<a href="#">Amazon CloudFront API Reference</a>	The API reference gives the schema location; complete descriptions of the API actions, parameters, and data types; and a list of errors that the service returns.
<a href="#">Amazon CloudFront Release Notes</a>	The release notes give a high-level overview of the current release. They specifically note any new features, corrections, and known issues.
<a href="#">Technical documentation for the Amazon Simple Storage Service (S3)</a>	The technical documentation provides a detailed discussion of the service. It includes the basics of getting started, an overview of the service, programming reference, and API reference.
<a href="#">AWS Developer Tools</a>	A central starting point to find documentation, code samples, release notes, and other information to help you build innovative applications with AWS.
<a href="#">AWS Management Console</a>	The console allows you to perform most of the functions of Amazon CloudFront without programming.
<a href="#">Discussion Forums</a>	A community-based forum for developers to discuss technical questions related to Amazon CloudFront.
<a href="#">AWS Support Center</a>	The home page for AWS Technical Support, including access to our Developer Forums, Technical FAQs, Service Status page, and Premium Support (if you are subscribed to this program).
<a href="#">AWS Premium Support Information</a>	The primary web page for information about AWS Premium Support, a one-on-one, fast-response support channel to help you build and run applications on AWS Infrastructure Services.
<a href="#">Amazon CloudFront product information</a>	The primary web page for information about Amazon CloudFront.

Resource	Description
<a href="#">Contact Us</a>	A central contact point for inquiries concerning AWS billing, account, events, abuse, etc.
<a href="#">Conditions of Use</a>	Detailed information about the copyright and trademark usage at Amazon.com and other topics.

## Where Do I Go from Here?

---

Although fairly simple to use, CloudFront is rich in functionality. There are a variety of resources you can use to learn more about CloudFront. In addition to the [Amazon CloudFront documentation](#), resources include AWS SDKs, feature summaries and links to third-party applications that work with CloudFront. The list below includes a few of these resources that you might find helpful.

### Topics

- [Amazon CloudFront SDKs \(p. 218\)](#)
- [Amazon CloudFront Getting Started Video \(p. 219\)](#)
- [Using Amazon CloudFront Logging \(p. 219\)](#)
- [Setting a Default Root Object \(p. 219\)](#)
- [Invalidating Objects \(p. 219\)](#)
- [Distributing Streaming Media \(p. 219\)](#)
- [Increasing Web Site Performance \(p. 220\)](#)
- [Creating Secure Connections Using HTTPS \(p. 220\)](#)
- [Distributing Private Content \(p. 220\)](#)
- [Using Custom Origins \(p. 220\)](#)
- [Amazon CloudFront Third-party Tools Overview \(p. 220\)](#)
- [Using CloudFront with a Content Management System \(p. 220\)](#)

## Amazon CloudFront SDKs

AWS offers SDKs that let you access CloudFront programmatically.

- [AWS SDK for Java](#)
- [AWS SDK for .NET](#)
- [AWS SDK for PHP](#)
- [AWS SDK for Ruby](#)

# Amazon CloudFront Getting Started Video

AWS: [Amazon CloudFront Getting Started](#)

## Using Amazon CloudFront Logging

AWS Blog: [Amazon CloudFront Request Logging](#) (for content delivered via HTTP)

AWS Blog: [Amazon CloudFront Now Supports Streaming Access Logs](#) (for content delivered via RTMP)

AWS Blog: [Enhanced CloudFront Logs, Now With Query Strings](#)

## Setting a Default Root Object

CloudBerry Lab: [How to set CloudFront Default Object with CloudBerry S3 Explorer](#)

## Invalidating Objects

In addition to the invalidation methods provided by CloudFront, you can use the following third-party tools to invalidate objects.

### Note

These tools were developed by third-party vendors who are not associated with Amazon Web Services. For information on how to use these tools, please refer to the vendor's documentation or contact the vendor.

- CloudBuddy Personal, <http://m1.mycloudbuddy.com/index.html>
- CloudBerry Explorer, <http://cloudberrylab.com>
- Ylastic, <http://ylastic.com>
- Cyberduck, <http://cyberduck.ch>
- Bucket Explorer, <http://www.bucketexplorer.com>
- CloudFront Invalidator, <http://www.swook.net/p/cloudfront-invalidator.html>
- CDN Planet CloudFront Purge Tool, <http://www.cdnplanet.com/tools/cloudfront-purge-tool/>

You can also search for code samples on Github, <https://github.com>. Search on the phrase "CloudFront invalidation".

## Distributing Streaming Media

AWS Articles & Tutorials:

- [Configuring Amazon CloudFront Streaming Using Adobe's Flash Builder](#)
- [Configuring Amazon CloudFront Streaming Using Flowplayer](#)
- [Configuring Amazon CloudFront Streaming Using JW Player](#)

StreamingMedia.com: [How To Get Started With Amazon CloudFront Streaming](#)

loncannon.net:

- [iPhone Windowed HTTP Live Streaming Using Amazon S3 and CloudFront Proof of Concept](#)
- [HTTP Live Video Stream Segmenter and Distributor](#)
- [iPhone Windowed HTTP Live Streaming Server](#)

Flowplayer.org: [Bandwidth detection: Make sure you reach your entire audience with good quality](#)

LongtailVideo.com (JW Player):

- [Using Amazon Web Services](#)
- [Using CloudFront](#)

## Increasing Web Site Performance

AWS Blog: [Improving Global Application Performance](#)

## Creating Secure Connections Using HTTPS

AWS Blog: [Amazon CloudFront: HTTPS Access, Another Edge Location, Price Reduction](#)

## Distributing Private Content

AWS Blog: [New Amazon CloudFront Feature: Private Content](#)

## Using Custom Origins

AWS Blog: [New Amazon CloudFront Feature: Custom Origins](#)

## Amazon CloudFront Third-party Tools Overview

AWS Blog: [CloudFront Management Tool Roundup](#)

## Using CloudFront with a Content Management System

### Drupal

- [Drupal.org: CloudFront Installation](#)
- [DrupalModules.com: CloudFront Drupal Module](#)



### **Sitecore**

- NTT Data Advisory Service: [AWS CloudFront Sitecore Integration](#)

### **WordPress**

- om4.com: [Using Amazon CloudFront with WordPress and WordPress MU](#)
- WordPress.org: [W3 Total Cache](#)
- WordPress.org: [Simple Amazon S3 Upload Form](#)
- WordPress.org: [OSSDL CDN Off-linker](#)
- WordPress.org: [My CDN](#)
- Inquisiter.com: [Amazon CloudFront CDN with a WordPress Blog](#)

# Document History

---

The following table describes the important changes to the documentation since the last release of CloudFront.

- **API Version:** 2012-05-05
- **Latest documentation date:** June 22, 2012

Change	Description	Date Changed
New Features	<p>This release of CloudFront introduces the following features:</p> <ul style="list-style-type: none"><li>• You can now invalidate objects using the CloudFront console. For more information, see <a href="#">Invalidating Objects (p. 68)</a>.</li><li>• The CloudFront console was updated to better support viewing on tablet devices.</li></ul>	June 22, 2012
New Features	<p>This release of CloudFront introduces the following features for download distributions:</p> <ul style="list-style-type: none"><li>• You can forward query strings to your origin. For more information, see <a href="#">How CloudFront Forwards and Logs Query String Parameters (p. 62)</a>.</li><li>• You can specify up to 10 origins. For more information, see <a href="#">Values that You Specify When You Create or Update a Download Distribution Using the CloudFront Console (p. 38)</a>.</li><li>• You can specify path patterns. For more information, see <a href="#">Values that You Specify When You Create or Update a Download Distribution Using the CloudFront Console (p. 38)</a>.</li></ul> <p>In addition, the CloudFront console has been updated. For more information, see <a href="#">Creating Download Distributions (p. 21)</a> and <a href="#">Creating Streaming Distributions (p. 28)</a>.</p> <p>The <i>Amazon CloudFront Getting Started Guide</i> was merged into the <i>Amazon CloudFront Developer Guide</i>, and the <i>Amazon CloudFront Developer Guide</i> was reorganized to enhance usability.</p>	May 13, 2012

Change	Description	Date Changed
Updated Documentation	The documentation about working with objects was reorganized and clarified. For the revised documentation, see <a href="#">Working with Objects</a> (p. 60).	April 4, 2012
New Documentation	Documentation about live streaming with Microsoft IIS Media Services version 4.1 was added. For more information, see <a href="#">Live Smooth Streaming Using Amazon CloudFront and IIS Media Services 4.1</a> (p. 174).	April 1, 2012
Updated Documentation	The documentation about live streaming with Adobe Flash Media Server was updated with information about Adobe Flash Media Server version 4.5. For more information, see <a href="#">Live HTTP Streaming Using CloudFront and Adobe Flash Media Server 4.5</a> (p. 152).	March 29, 2012
New Feature	<p>This release of CloudFront reduces the minimum TTL value for a download distribution. If you don't specify a minimum TTL when you create a distribution, CloudFront sets the minimum TTL to 0 seconds. For more information, go to the following documentation:</p> <ul style="list-style-type: none"> <li>• <a href="#">CloudFront product page</a></li> <li>• "Caching Duration and Minimum TTL" at <a href="#">Request and Response Behavior for Amazon S3 Origins</a> (p. 77)</li> <li>• "Caching Duration and Minimum TTL" at <a href="#">Request and Response Behavior, and Supported HTTP Status Codes for Custom Origins</a> (p. 79)</li> <li>• The <code>CachingBehavior</code> element in the <a href="#">DistributionConfig Complex Type</a>.</li> </ul>	March 15, 2012
Updated Documentation	Topics about live streaming with Adobe Flash Media Server and about geoblocking were moved from a separate document into the <a href="#">CloudFront Tutorials</a> (p. 152) chapter in this guide.	February 2, 2012
New Feature	<p>This release of CloudFront introduces AWS Management Console support for creating a distribution with a custom origin, restricting your distribution to HTTPS exclusively, and specifying a default root object. For more information, go to the <a href="#">Amazon CloudFront product page</a> or see any of the following topics in the <i>Amazon CloudFront Developer Guide</i>:</p> <ul style="list-style-type: none"> <li>• <a href="#">Creating Download Distributions</a> (p. 21)</li> <li>• <a href="#">Using an HTTPS Connection to Access Your Objects</a> (p. 126)</li> <li>• <a href="#">Specifying a Default Root Object (Download Distributions Only)</a> (p. 72)</li> </ul>	April 27, 2011
New Feature	This release of CloudFront includes integration with AWS Identity and Access Management (IAM). For more information, go to the <a href="#">Amazon CloudFront product page</a> or <a href="#">Controlling User Access to Your AWS Account</a> (p. 10) in the <i>Amazon CloudFront Developer Guide</i> .	March 10, 2011

Change	Description	Date Changed
New Feature	This release of CloudFront includes new APIs to support custom origins. For more information, go to the <a href="#">Amazon CloudFront product page</a> or <a href="#">Creating Download Distributions (p. 21)</a> in the <i>Amazon CloudFront Developer Guide</i> .	November 9, 2010
New Feature	This release of CloudFront includes new APIs for object invalidation. For more information, go to the <a href="#">Amazon CloudFront product page</a> or <a href="#">Invalidating Objects (p. 68)</a> in the <i>Amazon CloudFront Developer Guide</i> .	August 31, 2010
New Feature	CloudFront now supports the ability to assign a default root object to your distribution. For more information, see <a href="#">Specifying a Default Root Object (Download Distributions Only) (p. 72)</a> .	August 5, 2010
New Feature	Access logging for HTTP distributions now includes a field for query string parameters. For more information, see <a href="#">Download Distribution File Format (p. 133)</a> .	July 14, 2010
New Feature	Added support for secure connections using HTTPS. For more information, see <a href="#">Using an HTTPS Connection to Access Your Objects (p. 126)</a> .	June 7, 2010
New Feature	Added logging for streaming content. For more information, see <a href="#">Streaming Distribution Log File Format (p. 135)</a> .	May 13, 2010
New Feature	Reduced the minimum amount of time an object can be on an edge server from 24 hours to 1 hour. The default, however, remains 24 hours. For more information, see <a href="#">Expiration (p. 65)</a> .	April 13, 2010
New Feature	Added feature to serve private streaming content over a Real-Time Messaging Protocol (RTMP) and prevent the content from being downloaded. For more information, see <a href="#">Overall Process to Serve Private Content (p. 88)</a> .	March 28, 2010
New Feature	Added feature to deliver streaming content over a Real-Time Messaging Protocol (RTMP) connection. For more information, see <a href="#">Creating Streaming Distributions (p. 28)</a> .	December 15, 2009
New Feature	Added feature to restrict access to your content delivered over HTTP. For more information, see <a href="#">Using a Signed URL to Serve Private Content (p. 84)</a> .	November 11, 2009
New Guide	We've separated the API reference material into its own guide. The <i>Amazon CloudFront Developer Guide</i> contains general information about how to use CloudFront, and the <a href="#">Auto Scaling API Reference</a> contains detailed information about the control API requests, responses, and errors.	November 11, 2009