Content Distribution for Joomla!

jomCDN

# 

Support:

Tel: 269-979-5582

support@corephp.com

Table of Contents

Introduction 3

Terms to know 3

Requirements 4

Installation Guide 5

Joomla 1.6 Caching 5

Setting up Amazon S3 and CloudFront 5

Setting up a Cron Job 7

Considerations Regarding CloudFront 8

Debugging 8

Editing Files 9

# Introduction

A Content Delivery Network, or Content Distribution Network (CDN) is a system of computers that are placed at various points around the world. These computers contain copies of data. When clients around the world access the data, it is pulled from the nearest computer in the network that has a copy instead of being pulled from the original central server. This maximizes bandwidth and load speed and prevents bottlenecking so all clients around the world can quickly access the data without a problem. jomCDN is a Content Delivery Network specific to Joomla!. Below is a visualization of a Content Delivery Network.



## Terms to know

**Objects** – These can be a wide range of files including CSS, JavaScript, images, etc.

**Amazon S3** – This is a storage service that stores objects. These objects can be accessed through the web.

**Edge Locations** – These are locations of servers world-wide. The closest server to your location will respond to your request. This system improves website load speeds.

**CloudFront** –  This is a web service for content delivery that integrates with Amazon S3 to distribute objects through Edge Locations around the world.

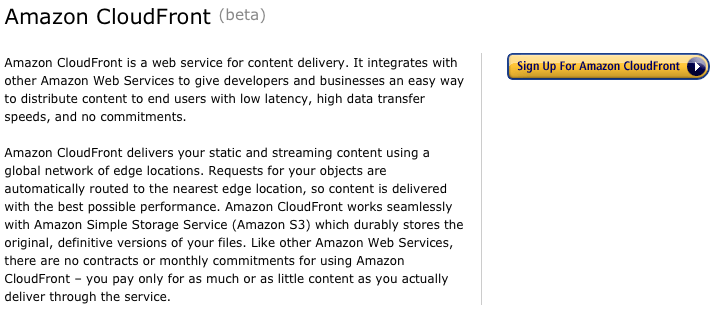
## Requirements

Before enabling these plug-ins, you must sign up for the following services:

* [Amazon S3 service](http://aws.amazon.com/s3/)



* (Optional/Recommended) [CloudFront](http://aws.amazon.com/cloudfront/) for a true CDN. We highly recommend utilizing CloudFront. Normally, when requesting an object from Amazon S3 directly, the request will take on average about 500-800ms depending on your location in the world and the location of the bucket. If you use CloudFront it will take an average of about 20-30ms, which is a significant difference.



* Server Requirements
  + PHP 5.2 or higher
  + MySQL 5.0.3 or higher

# Installation Guide

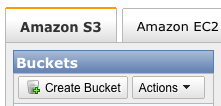
We have designed our instructions to prevent most problems you may encounter. Some installations may not go as smoothly as others, but if you follow these simple instructions step by step, you will be up and running in no time.

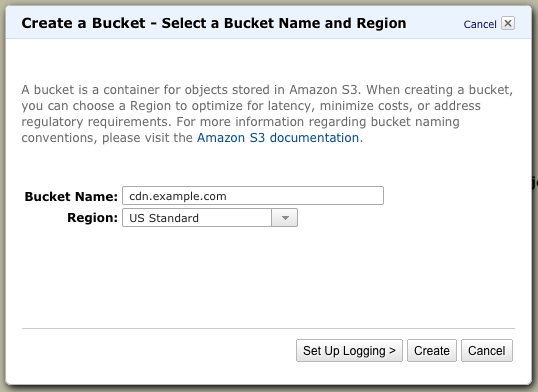
## Joomla 1.6 Caching

In order for jomCDN to be able to work with Joomla! 1.6, caching must be set to “Progressive” or “Conservative.” The setting can be modified by logging into the Joomla! back-end and going to Site->Global Configuration->System->Cache Settings. System caching does not need to be enabled for Joomla! 1.5.

## Setting up Amazon S3 and CloudFront

1. Sign up for the services listed above under “Requirements.”
2. Once you have created an account with Amazon S3, create a bucket through your AWS Management Console. You can create the bucket in any origin server you want. We recommend US – Standard or N. California. (See images below.)



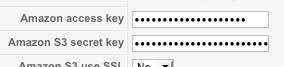


1. Once you have your bucket created, copy the name of the bucket to the plugin parameters on the right.

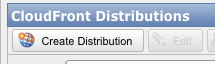
::Desktop:settings-bucket-name.png

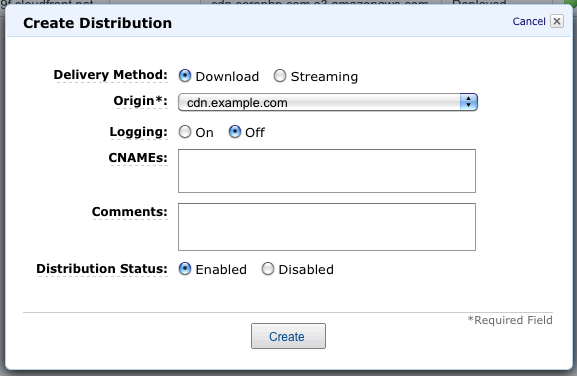
1. Now we need your Amazon credentials. These are unique to your AWS account. You can get these from the [Security Credentials](https://aws-portal.amazon.com/gp/aws/developer/account/index.html?ie=UTF8&action=access-key" \t "_blak) page. You will need your 'Access Key ID' and 'Secret Access Key.' Copy those Keys to the plugin parameters on the right.

::Desktop:aws-accesskey.png



1. At this point you can start using this plugin, although we HIGHLY recommend you set up [CloudFront](https://aws-portal.amazon.com/gp/aws/developer/subscription/index.html?productCode=AmazonCloudFront" \t "_blank). Once again, instead of object requests taking 500-800ms depending on location, CloudFront will increase the speed to an average of 20-30ms, which is a significant difference.
2. Setting up CloudFront (Optional/Recommended) - Once you have enabled CloudFront on your account, you need to create a Distribution from the [CloudFront Management Console](https://console.aws.amazon.com/cloudfront/home" \t "_blank). You will create the distribution for the bucket that you previously created, and the Delivery Method will need to be set to Download. (Setting up a CNAME is not recommended at this time, as it does not support https requests.) Once created, Amazon will create a unique Domain Name for your distribution. You will copy this domain name as is to the 'CloudFront' option in the plugin parameters.





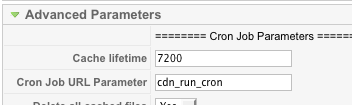
::Desktop:settings-cloudfront.png

## Setting up a Cron Job

You must set up a cron job. Note: for general information about setting up a cron job, see [Intro To Cron Jobs](http://www.unixgeeks.org/security/newbie/unix/cron-1.html). For information regarding your hosting provider, visit your hosting provider’s FAQ or manual pages.

**Note to Windows Users:** There are similar programs to cron available if your host runs Windows. [VisualCron](http://www.visualcron.com), [Cron for Windows](http://cronw.sourceforge.net/) and [pycron](http://www.google.com/url?sa=U&start=2&q=http://sourceforge.net/projects/pycron/&e=1102&mr=8,2%21619021,%21a:sr4) are all examples. Consult these projects' documentation for further information.

1. Change the Cron Job URL Parameter from the Advanced Parameters to something unique for your website. This cron job URL should be accessed at least once a day, but it all depends on the cache lifetimes that you set up.



1. Set up a UNIX cron job to have your server periodically view your cron job URL using a command-line HTTP agent like wget, curl or GET. The command to execute will look like this:

wget -N http://example.com/?cdn\_run\_cron

Where “cdn\_run\_cron” is the URL parameter that you set. If you use a different program than wget, substitute that program and its arguments for wget in this line.

1. After you have set up the cron job, you can enable the plugin!
2. Once the plugin is enabled, browse to your cron job URL manually, this will create the necessary database tables.
3. You can then browse through as many pages on your website as you can. This will cache all of the pages to the server.
4. When you are done, visit the cron job URL again. This will push all of the files over to the CDN. As files get pushed over, the new URLs of the files will be displayed when browsing your website. As time goes on, people that use your website will visit new pages, and new files will be found every time that the cron job runs. Those files will be pushed to the CDN and then the new URL of the file will be displayed in the source code of your HTML pages.

# Considerations Regarding CloudFront

When using your own domain name (CNAME), CloudFront does not support SSL. This plugin will automatically use SSL if a user is on an SSL page on your website. If you have SSL pages, don't use your own CNAME. Instead, use the one given for the distribution.

When using CloudFront, the edge locations will cache files, depending on their cache lifetime that you set on the parameters. Files will not be updated unless that lifetime expires, regardless of whether or not you update the file on S3. If you need to refresh single or multiple files see the “Editing Files” section.

# Debugging

If your template is not looking right and some images are missing, go to the Advanced Parameters of jomCDN and turn on the option “Delete all cached files”. Then, visit the cron job URL that you set up. This will clear the database from all cached files and push all the files over again. You will also have to create a new distribution if you are using CloudFront. This should rarely happen. If you still have problems, please contact us.

# Editing Files

Once a file is on the CDN, it won’t be pushed-over/updated until the cache for that file expires. For example, if you need to change a CSS file because you need to style a new part of your website, you should then rename that CSS file. (If you have a CSS file named styles.css, you could rename it to styles-20100801.css). This way, on the next cron job, this new CSS file will be pushed to the CDN and you won’t have to delete all of the cached files on the CDN.

If you have any support questions or suggestions, please contact us at [https://www.corephp.com/](https://www.corephp.com/members/supporttickets.php" \t "_blank)