Edgecast

HTTP Streaming

edgecast

Disclaimer

Care was taken in the creation of this guide. However, Edgecast cannot accept any responsibility for errors or omissions. There are no warranties, expressed or implied, including the warranty of merchantability or fitness for a particular purpose, accompanying this product.

Trademark Information

EDGECAST is a registered trademark of Edgecast Inc.

JW PLAYER is a registered trademark of LongTail Video.

FLOWPLAYER is a registered trademark of Flowplayer Ltd.

IOS, IPAD, APPLE TV, and SAFARI are registered trademarks of Apple Inc.

About This Guide

HTTP Streaming
Version 2.31
6/2/2022

Table of Contents

HTTP Streaming Solutions

Solution Comparison

We provide various solutions through which you can stream live and on-demand content over the HTTP Large platform. Each streaming solution supports a different feature set. A brief description is provided below for major streaming features.

- **Live Streaming:** Indicates the capacity to stream a live event as it occurs.
- **On-Demand Streaming:** Indicates the capacity to stream media as it is requested.
- **Progressive Streaming:** Indicates that a media player will request media in a progressive sequence from start to finish. Normally, this would prevent a user from jumping ahead (i.e., seeking) to a position in the video that has not yet been downloaded. However, our technology allows this capability under certain circumstances. For more information, please refer to the **Seeking within a Video** section of the **HTTP Progressive Download** chapter.
- **Dynamic Streaming:** Indicates that it has the capability to broadcast multiple streams of varying quality. This allows each client's media player to intelligently and dynamically choose the stream that best matches the client's available bandwidth and CPU processing power.
- **Supported Media Player(s):** Indicates the media players that are natively supported by the streaming solution. Keep in mind that this field is not meant to be exhaustive.

The following table indicates the features/players that each of these solutions natively support.

Name	Live Streaming	On-Demand Streaming	Dynamic Streaming	Supported Media Players
Dynamic Cloud Packaging	●	●	●	Standard HLS and DASH players.
HTTP Progressive Download		●		All*

Note: The HTTP Progressive Download streaming solution can be used with any media player that supports the requested media format (e.g., mp4, mov, etc.).

Note: Android 4.0 and 4.1+ provide limited native support for HTTP Live Streaming. View a sampling of the issues that your viewers may encounter.

Note: For more information on Dynamic Cloud Packaging, please refer to the CDN Help Center.

Reports & Analytics

CDN activity for our HTTP streaming solutions is tracked under the HTTP Large platform. Each module in the Analytics Suite allows you to generate reports on the HTTP Large platform. These reports will contain data for these streaming solutions and standard traffic on the HTTP Large platform.

HTTP Progressive Download

Introduction

HTTP Progressive Download uses the HTTP protocol to stream archived media content. As a result, it cannot take advantage of the security features that are available with the RTMP protocol, such as encryption and SWF verification. Additionally, the entire video is downloaded to a user's computer. As a result of all of these factors, HTTP Progressive Download is more susceptible to piracy than some of our other streaming solutions.

As previously mentioned, the HTTP Progressive Download uses the HTTP protocol to stream video on-demand. Specifically, HTTP Progressive Download should only be used with the HTTP Large platform. This allows progressively downloaded content to take advantage of all of the features provided by the HTTP Large platform. For example, you can use an existing customer origin to indicate the location of your video content.

Note: Any type of asset supported by your video player can be streamed through HTTP Progressive Download. However, seeking within a video is only supported for H.264 encoded video.

HTTP Progressive Download Requirements

HTTP Progressive Download requires an origin server, our content delivery network, and a video player. The requirements for each of these items are listed below.

System Requirements:

- **Storage:** CDN Storage (i.e., CDN Origin Server) or Customer Origin Server
- **Content Delivery Network:** We provide servers that transmit your on-demand content to all necessary edge servers for delivery to your end-users.
- **Video Player:** Make sure that your video player supports the type of media that you would like to stream. No additional requirements are needed for HTTP Progressive Download.

Setting up HTTP Progressive Download

Setting up HTTP Progressive Download involves the following steps:

1. Upload your media content to either CDN or a customer origin server.
2. Point your media player to the appropriate player URL. Make sure to use either a CDN or edge CNAME URL.

Note: For additional information on how to setup an origin server and upload content, please refer to the **Configuring an Origin Server** chapter.

Media Player

Once the desired media content is hosted on a customer or CDN origin server, you are ready to set up your media player for use with HTTP Progressive Download. This requires that you set the player URL to a CDN or edge CNAME URL that leverages the HTTP Large platform (as indicated below).

- **CDN URL:** http://wpc.*xxxx*.edgecastcdn.net/*yyxxxx*/*Path*/*Filename*
- **Edge CNAME URL:** http://*cname.hostname.com*/*Path*/*Filename*

When defining a CDN URL, you should keep the following items in mind:

- **Account number:** The term *xxxx* should be replaced with your MCC account number. This account number can be found in the upper-right hand corner of the MCC.
- **Origin identifier:** The term *yy* should be replaced with one of the following:
  - **Customer origin server:** It should be replaced with "80" (e.g., /800001).
  - **CDN origin server:** It should be replaced with "00" (e.g., /000001).
- **Path:** The term *Path* should be replaced by the path to the folder where the desired media content is stored.
  - **Customer origin server:** The specified path should start with the name of the customer origin configuration that points to your origin server.
- **Filename:** The term *Filename* should be replaced by the filename of the desired media asset (e.g., video.mp4).
- **Token-Based Authentication:** If the desired video has been secured by Token-Based Authentication, then you will need to append to the filename a question mark (?) followed by a token value that corresponds to an encryption key. For example, if the filename is "video.mp4" and the desired token is "a4fbc3710fd3449a7c99984d1b86603c22be1006d830b," then you would specify "video.mp4?a4fbc3710fd3449a7c99984d1b86603c22be1006d830b" as your filename.

For more information, please refer to the **Token-Based Authentication User Guide**, which is available from the Media Control Center (MCC).

Tip: For more information on edge CNAME URLs, please refer to the **Masking the URL of a CDN Origin Server (CNAME)** section below.

Seeking Within a Video

HTTP Progressive Download provides seeking functionality for H.264 encoded videos (i.e., MP4, F4V, and MOV). This functionality allows clients to seek to a particular position in the video, regardless of whether the video has already been cached by the user agent (i.e., web browser) that requested it. Additionally, it allows you to programmatically start a video at a particular point in time.

Important: If you would like to seek within an H.264 encoded (e.g., MP4) asset, then the edge server must have the entire asset in cache, or an H.264 streaming module must be installed on the origin server and it must be configured to support ec_seek for H.264 encoded assets. The ec_seek functionality has been properly configured on our CDN origin servers for MP4 assets. For all other H.264 file formats, please contact your CDN account manager to activate ec_seek functionality on a CDN origin server.

Important: Keep in mind that unique query string caching will cache each seek request. It is recommended that you limit unique query string caching to those folders where dynamic content is generated. By excluding folders that contain media content, your clients will be able to seek without caching multiple versions of the same asset. For more information on how to customize your unique query string caching configuration, please contact your CDN account manager.

A video player can be implemented with a scrubber bar (a.k.a. seek bar). A user can jump to a particular position in the video by clicking on the desired position in the scrubber bar. However, before you can take advantage of this functionality you will need to provide a hint to your video player to use "ec_seek" instead of its native parameter. Below you will find an example of how to implement this configuration for JW Player.

- **JW Player 5.x (5.3 and above):** 'http.startparam': 'ec_seek'
- **JW Player 5.1:** http.startparam=ec_seek

Note: For information on how to configure JW Player 5.x, please refer to the documentation that was provided with it.

Note: The "ec_seek" parameter is unnecessary when using JW Player 6.

Note: If you would like to configure a different video player, keep in mind that the seek parameter (e.g., http.startparam) may be different. Please refer to that documentation provided with the desired player.

Once you have configured your video player to use the "ec_seek" parameter, a user can jump ahead to portions of the video that have not yet been downloaded to his/her computer. This can be performed using a player-specific seeking parameter. For example, JW Player uses "start" as can be seen in the following code fragment:

```
file: "http://wpc.0001.edgecastcdn.net/000001/Video01.mp4",
'http.startparam': 'ec_seek',
start: 30
```

Note: The above sample code assumes that the requested video was encoded using H.264 format. It will seek 30 seconds into an H.264 encoded video.

Note: The above scenario assumes that a user chose to skip ahead to a portion of the video that had not been already downloaded to his/her computer. However, you may also configure a link to always start at a certain position. In such a case, our CDN will always generate a truncated video asset that starts at the specified position.

Setting the Start Position for H.264 Encoded Videos

You can programmatically determine the position at which an H.264 encoded video (i.e., MP4, MOV, and F4V) will be started by setting the ec_seek parameter. This parameter will need to be set to the number of seconds elapsed from the beginning of the video.

In order to calculate the exact position where a video will start, you will need to use the following formula:

ec_seek =*Seconds*

Providing Seek Support for a Custom Player

A custom player must comply with the following specifications in order to implement seek support:

- The seek parameter must be defined as "ec_seek."
- The ec_seek parameter must be implemented as a query string parameter. Each seek request generated by the player must contain an ec_seek query string parameter.
- The ec_seek parameter must be defined as indicated above for either H.264 videos.
- If you plan on seeking on videos that have been protected by Token-Based Authentication, you will need to ensure that the ec_seek parameter is appended to the token value. A token value must be the first query string parameter in the requested URL (e.g., http://www.mydomain.com/video.mp4?ak32d0kd0fm12fonmf2&ec_seek=0).