EZDRM Configuration
Apple FairPlay License Delivery

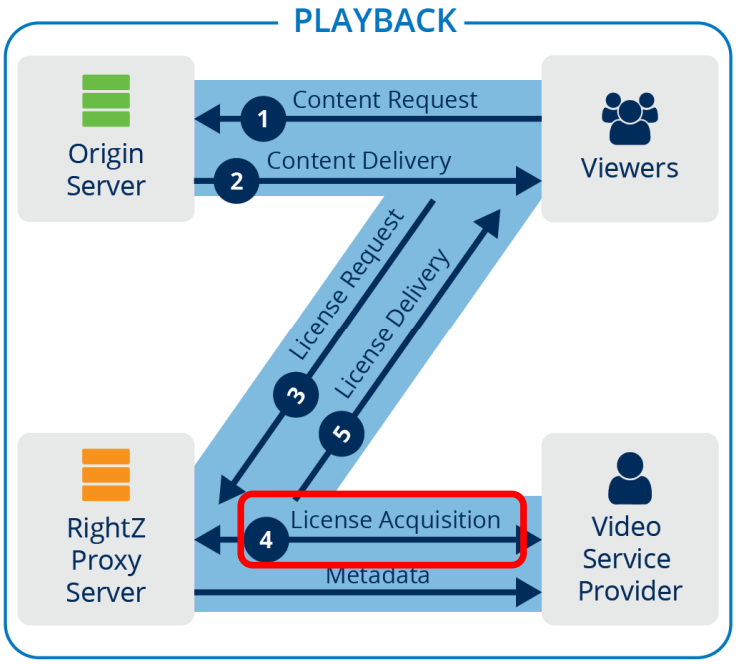
Table of Contents

Version 1.3 – updated Feb 9, 2022

Apple FairPlay DRM Setup Overview

EZDRM Apple FairPlay DRM is a hosted Apple FairPlay Streaming (DRM). This enables a content owner to encrypt their media with Apple FPS DRM keys and deliver content to Apple devices with native support in the MAC Safari browser via HTML 5 player or iOS via native App.

The EZDRM solution is composed of two separate processes. The first part is the encryption of media. This is called the packaging process. This is accomplished via a secure web call to the EZDRM KeyZ API. The KeyZ API will return an XML response with the DRM key structure. These encryption key values are what will be used by any compliant 3rd party encoder and packager application to DRM protect the content. These are outlined in our packaging documents for Wowza, Bitmovin, Shaka Packager, Bento 4 Open Source and others. Please check the Resources > Documentation tab of our site.



The second part of the process is the issue of a DRM license for the media asset for playback. This document outlines the Playback and License acquisition process, as highlighted in the Playback figure above in step #4.

The second part of the process is the issue of a DRM license for the media asset. When the media player loads the asset, the FPS compliant player will read the HLS manifest and call the EZDRM license servers, also known as the RightZ API. The EZDRM license servers will open the request and match it to your account. Then the EZDRM servers will call your Authentication URL that is hosted on your web infrastructure. This call will pass the metadata of the player along with any additional custom data, such as session details, user identification or any custom details that are needed for your business logic to run. Upon receiving this request, your side will process any business logic that you need, and either return the DRM rules you wish EZDRM to issue or deny the request.

Playback License Delivery

Licenses are issued via an Authentication URL call-back method, where the player will send a DRM license request to EZDRM servers. EZDRM will then make a back-end web call to the client Authorization URL. This URL is stored in your account settings and is hosted on your web infrastructure. You will use this to process any business logic you may need to either grant licenses via returning license values or deny the request by dropping it.

Persistent vs. Non-Persistent Licenses

There are two types of licenses you can issue: persistent and non-persistent.

Persistent Licenses – license data is stored locally on the device and has a persistent location, where the license is stored. Repeated accessing of the media will not result in a new license call. If the license expires or becomes invalid, it will go through the licensing process again and store the new license data. Examples of browsers that support persistent licenses include Microsoft Internet Explorer (IE) and Edge on Windows computers. Android devices, as well as apps, such as Netflix and Google Play, also support persistent licenses because they offer a persistence license locker for storage. Changing devices or settings can result in a new license call and a new playback license.

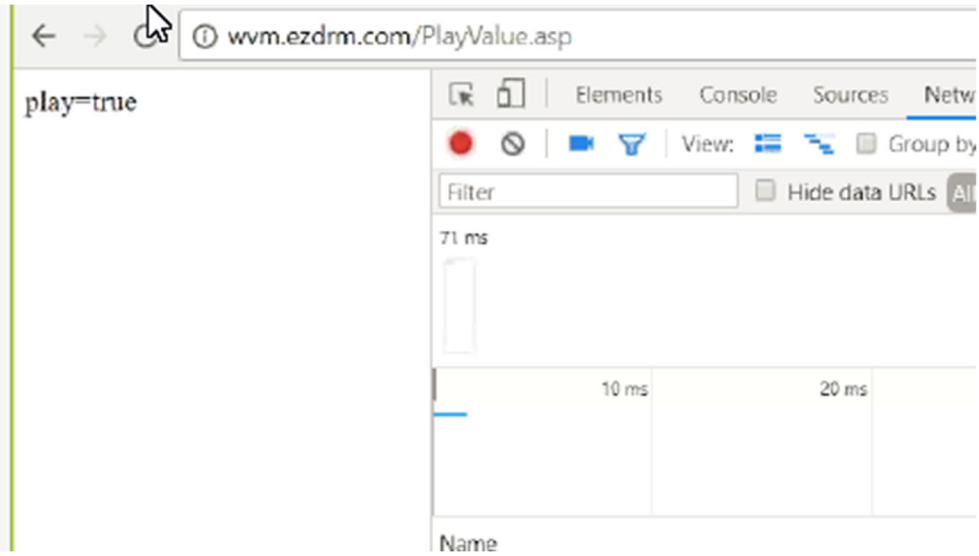
Non-Persistent Licenses – many browsers do not have a secure place to store license data. Each time the media is accessed, a new license call will be made resulting in a new non-persistent playback license. Examples of browsers that do not support persistent licenses are Firefox, Chrome and HTML5 players. If you are using

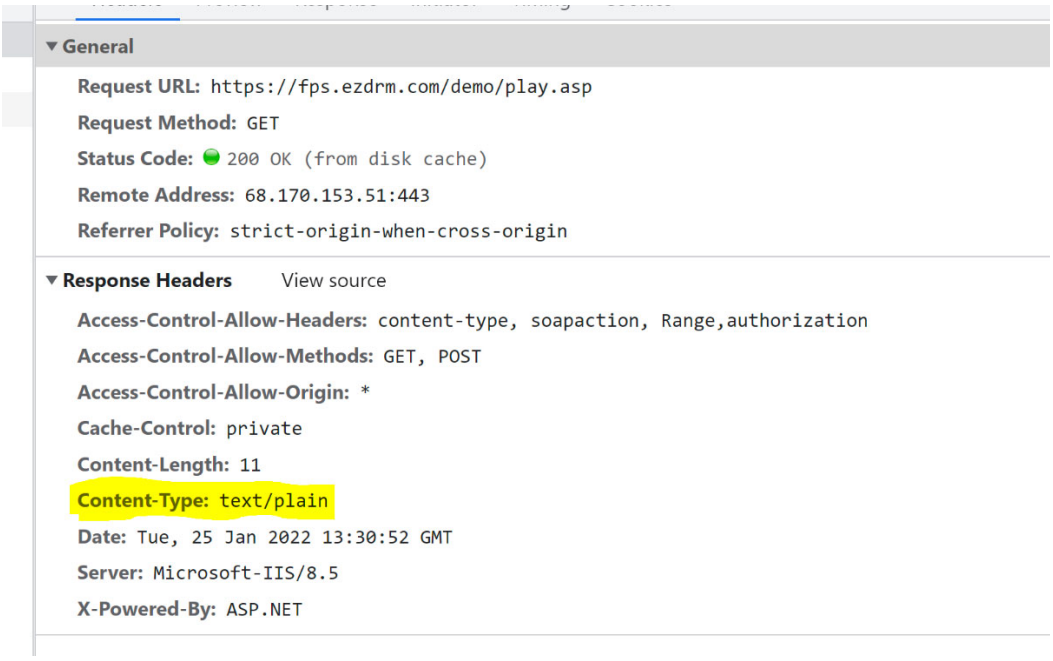
a browser, EZDRM will assume that you are requesting a non-persistent license because not all browsers are supported. For instance, Widevine does not currently support license persistence on Chrome or Firefox, although that could change in the future.

Apple FairPlay License Delivery

This generic authentication URL is a simple example of the FairPlay response: http://fps.ezdrm.com/demo/play.asp

FairPlay responses need be in text format (not in html format). The Authorization URL can be a dynamic URL or a flat TXT file, but both return back values in TXT format. Be sure that the response **Content-Type** is set to **text/plain** format.





Simple and Detailed Responses

Simple Response

The simple response **Play = True** assumes the following values:

```
Offline=False
Play=True
Rental_Duration=0
```

When using a browser, or other media that does not support persistent licenses, you can send the response **play=true** and EZDRM assumes the default settings above. There would be no need to return any additional values.

The descriptions of these values are as follows:

Parameter	Description	Value Type
Offline	Allows a persistent or non-persistent license; True=persistent license; False=no persistent license.	True/False
Play	Gives the media permission to play	True/False
Rental_Duration	Time window in seconds that playback is permitted, a value of 0 indicates that there is no limit to the duration. **Mutually exclusive to Lease_Duration**	Seconds; default is 0
Lease_Duration	The duration of the lease. Used in Live streaming. **Mutually exclusive to Rental_Duration**	Seconds; default is 0
airplay	Content will be sent by AirPlay to an Apple TV box.	True/False; Default is False

HDCP	Determines whether High-bandwidth Digital Content Protection (HDCP) enforcement is required for SD, HD, UHD1, or UHD2 tracks.	**Not required:** HDCP is not enforced. **Type 0 required:** HDCP type 0 content streams is enforced. This allows the stream to be transmitted by the HDCP repeater to all HDCP devices. **Type 1 required:** HDCP content type 1 (HDCP version 2.2 or later) is enforced. Streams may not be transmitted by the HDCP repeater to HDCP 1.x-compliant devices or HDCP2.0-compliant repeaters.
Adapter	Permission to output to external non-HDMI screen	True / False Default is True
Storage_duration	This specifies the maximum time the key stays valid prior to playback being started. Measured from license acquisition time.	Seconds; default is 0

Detailed Response

When using an app, or both a browser and an app you can send a more detailed license response, including requesting a persistent license. If a detailed response is returned, EZDRM will return back those explicit attributes.

If you issue a persistent license to a device or browser that does not support persistent licenses, it will automatically be ignored and treated like a non-persistent license.

Sending a detailed response will allow you to set each specific attribute like the example below:

```
Offline=True
Play=True
Rental_Duration=6000
```

This response is for a license that lasts for 10 Minutes and will be stored locally on the client's device.

Passing Custom Data

As part of the licensing process, the content owner can pass a set of parameters through the EZDRM system in order to use that information within the business logic. These parameters should be attached to the server URL provided above.

This is an example in the HTML player:

```
var serverProcessSPCPath  = '<<FPS PATH TO LICENSE SERVER>>/api/licenses/<<ASSET ID>>?<<KEY1>>=<<VALUE1>>&<<KEY2>>=<<VALUE2>>';
var serverProcessSPCPath  = 'http://fps.ezdrm.com/api/licenses/09cc0377-6dd4-40cb-b09d-XXXXXXXX?CustomValue=Value1?CustomValue2=Value2';
```

The EZDRM system passes several values by default to your Authorization URL:

- The FPS License Server Path
- AssetID – changes per the asset; the AssetID is generated during DRM key generation.
- Custom Data values
- The Client IP of the end client is added to the end of the string

Here is an example of the EZDRM return post to your Authorization URL:

http://fps.ezdrm.com/api/licenses/09cc0377-6dd4-40cb-b09d-XXXXXXXX?customdata1=123?customdata2=345&ClientIP=12.34.567.89

Sample Player

For an iOS offline example, you can download using this link: www.ezdrm.com/downloads/samplePlayer-v4.zip

To use the player for testing you'll need to:

- Edit the Cert file
- Edit the m3u8