# EZDRM Configuration
# Universal DRM License Delivery

# Table of Contents

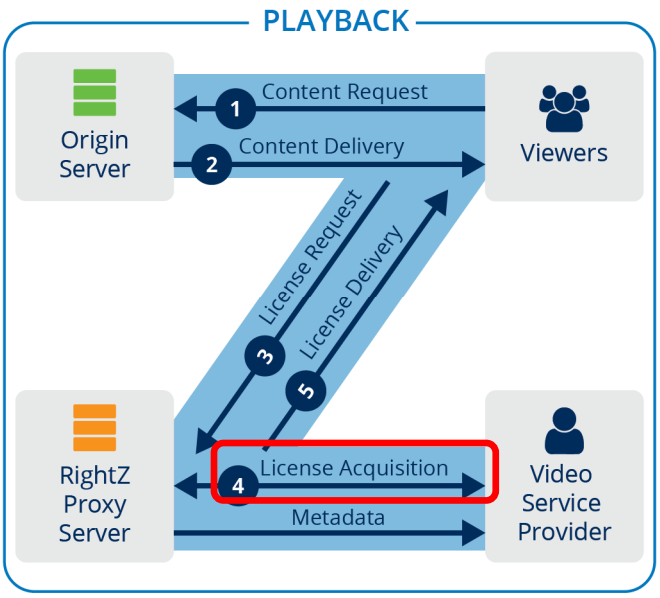


Version 1.1 / April 2021

# EZDRM DRM Setup Overview

EZDRM Universal DRM is a combination of Google Widevine Modular with Microsoft PlayReady, both using linked CENC keys over DASH streaming. This enables a content owner to encrypt the media once with CENC keys and deliver either a PlayReady License or a Widevine License depending on the player and platform calling for a license.

The EZDRM solution is composed of two separate processes. The first is encrypting your media, this is call the packaging process. This is accomplished via a secure web call to the EZDRM Key Servers API. The Key server 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. These are outlined in our packaging documents for Wowza, Bitmovin, MP4Box, and Bento 4 Open Source, etc.




**The next 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 above in Playback step #4.**

When the media player loads the asset, the DASH compliant player will read the manifest and automatically call the EZDRM license servers with the license request. The EZDRM license servers will then open the request and match it to your account. Then the EZDRM servers will call an Authentication URL (also referred to as Auth URL) that is hosted on your web infrastructure. This call will pass the meta data of the player along with any additional custom data such as session details, user identification or any custom details that is needed for your business logic to run. Upon receiving this request, you can process any business logic, 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 hosted on your web infrastructure and you will use this to process the business logic to grant licenses via returning license values.

The authentication URL is divided such as there is one URL for Widevine related license calls and one for PlayReady license calls. This is due to the fact that the underlying DRM technologies support different functionalities and your response needs to be formatted differently. You can use one authentication URL, if you wish to add advanced logic.

Upon creation of your account you will receive a default Authorization URL for use during your testing phases. To verify playback, you can point to the predefined test Authentication URL. Once testing is confirmed, the next phase is to modify your Authentication URL so that you can control the DRM licensing process.

## Persistent vs. Non-Persistent Licenses

There are two types of licenses issued; persistent and non-persistent.

**Persistent Licenses** – license data is stored locally on the device and has a persistent location where the license stored. Repeated accessing of the media will not result in a new license call. If the license becomes expired or 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. 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 persistent 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 Chrome or Firefox, although that could change in the future.

# Widevine License Delivery

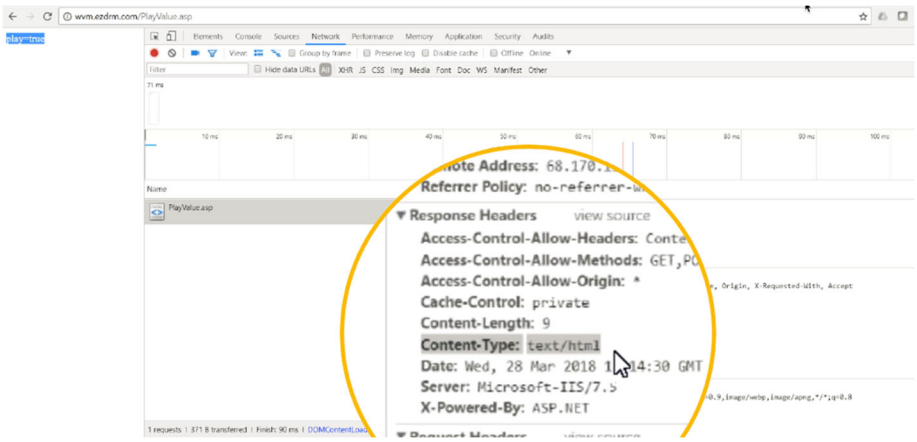
## Widevine Auth URL

This generic authentication URL is a simple example of the Widevine response: http://wvm.ezdrm.com/PlayValue.asp

This is an example of a simple response, which returns play=true.


Widevine responses need be in text format (not in html format). The Widevine 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/html** format.

***Note:*** *Each return value needs to be on its own line.*




EZDRM also returns the full Widevine call to your authorization URL for you to use within your business logic. This call is formatted with the Widevine information, followed by a line break and your user account PX value and any Custom data.

## Widevine Simple and Detailed Responses

### Simple response

The simple response **play = true** assumes the following values for Widevine:

```
Play=True
Persist=False
Rental_Duration=0
License_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.

### 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.

The descriptions of the Widevine values are as follows:

| Parameter | Description | Value Type |
|---|---|---|
| Play | Gives the media permission to play | True/False |
| Persist | Issues a persistent or non-persistent license. For Offline, persist must be set to true. | True/False; default is False |
| Rental_Duration | Time window in seconds that playback is permitted, a value of 0 indicates that there is no limit to the duration | Seconds; default is 0 |
| License_Duration | Time window in seconds for this specific license, a value of 0 indicates that there is no limit to the duration | Seconds; default is 0 |
| Playback_Duration_Seconds | Indicates a viewing window. Once playback starts, the license will expire after this number of seconds | Seconds; default is 0 |
| Output_Protection_Level | *string - one of:* HDCP_NONE, HDCP_V1, HDCP_V2, HDCP_V2_1, HDCP_V2_2, HDCP_NO_DIGITAL_OUTPUT | Default is HDCP_NONE |
| Security_Level | Defines client robustness requirements for playback.<br><br>1 - Software-based whitebox crypto is required. (SW_SECURE_CRYPTO)<br><br>2 - Software crypto and an obfuscated decoder is required. (SW_SECURE_DECODE)<br><br>3 - The key material and crypto operations must be performed within a hardware backed trusted execution environment. (HW_SECURE_CRYPTO)<br><br>4 - The crypto and decoding of content must be performed within a hardware backed trusted execution environment. (HW_SECURE_DECODE)<br><br>5 - The crypto, decoding and all handling of the media (compressed and uncompressed) must be | Default is 1 |

| | | |
|---|---|---|
| | handled within a hardware backed trusted execution environment. (HW_SECURE_ALL) | |
| Can_Renew | Indicates that renewal of this license is allowed. If true, the duration of the license can be extended by heartbeat. Default is false. | True/False |
| Renewal_Delay_Seconds | Indicates how many seconds after license_start_time before renewal is first attempted. **This field is used only if can_renew is true.** | Seconds |
| Renewal_Retry_Interval_Seconds | Specifies the delay in seconds between subsequent license renewal requests, in case of failure. **This field is used only if can_renew is true.** | Seconds |
| Renewal_Recovery_Duration_Seconds | The window of time in which playback can continue while renewal is attempted, A value of 0 indicates that there is no limit to the duration. **This field is used only if can_renew is true.** | Seconds |
| Renewal_Server_Url | This is the Widevine Proxy URL as per your EZDRM account | URL |

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

```
Play=True
Persist=True
Rental Duration=0
License_Duration=600
Playback_Duration_Seconds=0
Output_Protection_Level=HDCP_NONE
Security_Level=1
```

The above response is for a license that lasts for 10 minutes (600 seconds) and will be stored locally on the client's device.

## Widevine Server URL (aka Proxy URL)

## PX value

The **PX value** is branded for your account and is account specific. Your PX value is always the same no matter what packager you are using.

When generating DRM keys, the EZDRM values returned for **Widevine ServerURL** include the PX number at the end of the Proxy URL (the PX is the last six characters).


It is also the last six characters of your EZDRM Profile ID, as shown in this example:

Your PlayReady DRM Profile ID is: **4BE3XX9X-XX52-5RT5-87XX-34XXXX123456**

## Widevine Proxy URL for AWS MediaConvert / MediaLive

The Widevine authentication URL is the same for all packagers except for AWS. AWS has a different license proxy URL but the PX value is always the same value assigned to your account.

https://widevine-dash.ezdrm.com/widevine-php/widevine-foreignkey.php?pX=D5XX4X

## 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. The server URL will include your PX value and all of the other values should follow:

https://widevine-dash.ezdrm.com/proxy?pX=YOURPX&CustomData=123

## Widevine Metadata

The full Widevine call looks like this:

https://wvm.ezdrm.com/PlayValue.php?**p1=1**&response=%7b%22**status**%22%3a%22OK%22%2c%22license_metadata%22%3a%7b%22**content_id**%XX%**3a**%22**XXXXXXXXXX6Kf8ky**%**5c**%**2fHWXXX**%**3d**%**3d**%22%**2c**%22**license_type**%22%3a%22**STREAMING**%22%2c%22request_type%22%3a%22NEW%22%7d%2c%22**supported_tracks**%22%3a%5b%7b%22**type**%22%3a%22**SD**%22%2c%22key_id%22%3a%22As8OXXXXXXXXXXXXXbm%5c%2fg%3d%3d%22%7d%2c%7b%22type%22%3a%22AUDIO%22%2c%22**key_id**%22%**3a**%**228XXXXXXXXXX5DrbjTOXXXX**%3d%3d%22%7d%2c%7b%22**type**%22%3a%22**HD**%22%2c%22key_id%22%3a%22swXXXXXXXXXXXXXmR36Fg%3d%3d%22%7d%5d%2c%22make%22%3a%22Google%22%2c%22**model**%22%3a%22**ChromeCDM-Windows**%22%2c%22**security_level**%22%3a**3**%2c%22internal_status%22%3a0%2c%22session_state%22%3a%7b%22license_id%22%3a%7b%22request_id%22%3a%22ozRFujx7751+rsP0xFghIg%3d%3d%22%2c%22session_id%22%3a%22ozRFujx7751+rsP0xFghIg%3d%3d%22%2c%22purchase_id%22%3a%22%22%2c%22type%22%3a%22STREAMING%22%2c%22version%22%3a0%7d%2c%22signing_key%22%3a%22cS4LkUh%5c%XXXXXXXXXXXXXXXXXXXXXXXXXXVMwmoXXXXXXXXXXXXXXXXXXXXXXXXPHp1+WqZTUpbWPWHDHyuKlFnw%3d%3d%22%2c%22keybox_system_id%22%3a7650%2c%22license_counter%22%3a0%7d%2c%22drm_cert_serial_number%22%3a%22MTk4ZmFkYTA5MDkzOTc1ZDc2OGMxNjQ4ZWEyNDNkMmQ%3d%22%2c%22device_whitelist_state%22%3a%22DEVICE_NOT_WHITELISTED%22%2c%22**platform**%22%3a%22**pc**%22%2c%22device_state%22%3a%22RELEASED%22%2c%22pssh_data%22%3a%7b%22key_id%22%3a%5b%22As8OBPq2WWCESeh4ulbm%5c%2fg%3d%3d%22%5d%2c%22**content_id**%XX%**3a**%22**XXXXXXXXXX6Kf8ky**%**5c**%**2fHWXXX**%**3d**%**3d**%22%**2c**%22**client_max_hdcp_version**%22%3a%22**HDCP_V1**%22%2c%22client_info%22%3a%5b%7b%22name%22%3a%22architecture_name%22%2c%22value%22%3a%22x86-32%22%7d%2c%7b%22name%22%3a%22company_name%22%2c%22value%22%3a%22Google%22%7d%2c%7b%22name%22%3a%22**model_name**%22%2c%22value%22%3a%22**ChromeCDM**%22%7d%2c%7b%22name%22%3a%22**platform_name**%22%2c%22**value**%22%3a%22**Windows**%22%7d%2c%7b%22name%22%3a%22widevine_cdm_version%22%2c%22value%22%3a%221.4.9.1070%22%7d%5d%2c%22**platform_verification_status**%22%3a%22**PLATFORM_SOFTWARE_VERIFIED**%22%2c%22content_owner%22%3a%22movidone%22%2c%22content_provider%22%3a%22movidone%22%2c%22**client_ip**%22%3a%2**XXX.9X.X6.1XX**%22%7d&**pX=YOURPX**&**CustomData=123**

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

- The IP of the end client
- PX value, this is your EZDRM specific PX value for your account
- Custom Data is added to the end of the string

## Widevine Metadata Field descriptions

While most of these fields are only used by the license servers, the Custom Data values are passed by you though the system back to your Authorization URL for your business logic.

You may also want to record additional details that are in the base set of fields such as:

```
Model ChromeCDM-Windows-x86
```

This will list either if it was played on a PC such as Chrome or via an Android device.

The complete list is below, most of these values are specific to the Widevine System only.

| Field | Data | Description |
|---|---|---|
| p1 | 1 | Generic attribute from EZDRM |
| content_id | 3aXXXXXXXXX6Kf8ky5c2fHWXXX3d3d2c | The internal ContentID to the DRM system. Sending a Content_ID will allow you to encrypt content with the same DRM values as other content and have that content share one license |
| license_type | STREAMING | |
| supported_tracks | [{"type":"SD","key_id":"CUrXXXXXXXXWyihXXXXXX=="},{"type":"AUDIO","key_id":"\/TtI0XXXXXXXXXXXXXw=="},{"type":"HD","key_id":"uXXXXEd7VWOXXXXXXXXX==}] | Supported tracks and Key IDs |
| key_id | 3a228XXXXXXXXX5DrbjTOXXXX | |
| type | HD | |
| model | ChromeCDM-Windows | Model of the device making the license request |

| | | |
|---|---|---|
| security_level | 3 | Hardware device level of encryption (Level 1, 2 and 3) |
| platform | pc | Platform type |
| client_max_hdcp_version | HDCP_V1 | |
| model_name | ChromeCDM | |
| platform_verification_status | PLATFORM SOFTWARE VERIFIED | Values:<br>PLATFORM_UNVERFIED<br>PLATFORM_TAMPERED<br>PLATFORM_SOFTWARE_VERIFIED<br>PLATFORM_HARDWARE_VERIFIED |
| client_ip | XXX.9X.X6.1XX | IP Address for the client |
| playback | True | True or False |
| persist | False | True or False |
| rental_duration | 0 | Time window in seconds that playback is permitted, a value of 0 indicates that there is no limit to the duration |
| license_duration | 0 | Time window in seconds for this specific license, a value of 0 indicates that there is no limit to the duration |
| pX | Your EZDRM Account | Your EZDRM account PX |
| timestamp | 2016-09-14 14:54:56 | |

## Widevine Sample Players

For an offline sample player you can use the ExoPlayer. Download using this link:
http://www.ezdrm.com/downloads/ExoPersistent.zip

For an online sample player you can use the ExoPlayer. Download using this link:
http://www.ezdrm.com/downloads/ExoOnline.zip

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

- Edit the PX to your PX value
- Edit the mpd URL
- Password to unzip is "ezdrm"

For an online example use the Shaka Player found here:
https://shaka-player-demo.appspot.com/demo/#lang=en-US;build=uncompiled

# PlayReady License Delivery

## PlayReady Auth URL

This generic authentication URL is a simple example of the PlayReady URL response:
http://www.ezdrm.com/demo/SilverLightDRM-OOB-Out-of-Browser/postbackurl.asp

p1=5&p2=&p3=&p4=1&p5=0&p6=1&p7=0&p8=0&token=&CustomData=

## PlayReady Sample responses

Parameter (p) values are returned in the URL response. For example:

```
dim a as string = "p1=5&p2=&p3=&p4=1&p5=0&p6=1&p7=0&p8=0&token="
Response.Write(a & Request.QueryString("token")) Response.Write("&CustomData=" & Request.QueryString("CustomData"))
```

The parameters are as follows:

| Parameter | Description | Value Type |
|---|---|---|
| p1 – p3 | EZDRM values; ignore these values. | |
| p4=licensetype | Identifies if the license is persistent or non-persistent: 0 for non-persistent 1 for persistent | 0 or 1; default is 0 |
| p5=begindate | The license will be issued on a future begin date; currentdate + X (p5=1 the license will be valid 1 day after the current date) | Days (only valid if p4=1) |
| p6=enddate | License expiration in days (p6=1 issues a 1-day license) | Days (only valid if p4=1) |

| | | |
|---|---|---|
| p7=graceperiod | The grace period past the enddate that the license is still available; enddate + X | Days<br>(only valid if p4=1) |
| p8=opl | Output protection layer to block certain types of outputs | See Output Protection layer values in the Note below |
| p9=expiration | License expiration in seconds; (p9=60 issues a 60 second license) | Seconds |

***Note:*** *the list of allowed values for* ***p8=opl*** *are as follows:*

Output Protection Levels (OPLs)

| **Content Type** | **Allowed Values** |
|---|---|
| Minimum Compressed Digital Audio | 100, 150, 200, 250, 300 |
| Minimum Uncompressed Digital Audio | 100, 150, 200, 250, 300 |
| Minimum Compressed Digital Video | 400, 500 |
| Minimum Uncompressed Digital Video | 100, 250, 270, 300 |
| Minimum Analog Television | 100, 150, 200 |

An example of a PlayReady response for an offline one-day persistent license would be:

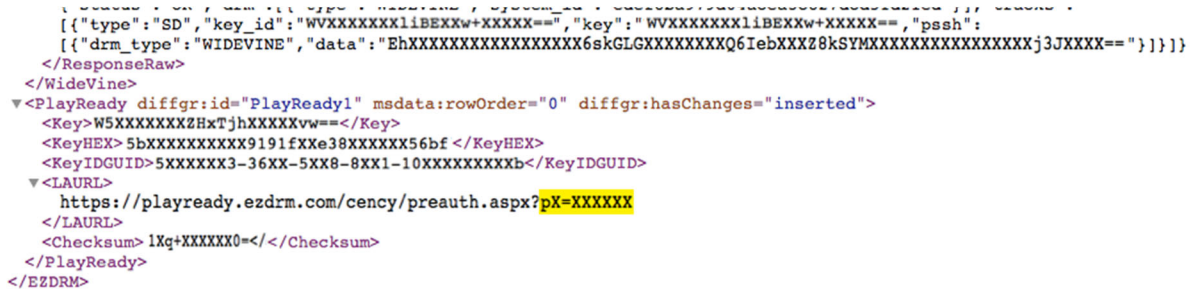
```
dim a as string = "p1=5&p2=&p3=&p4=1&p5=&p6=1&p7=0&p8=0&token="
Response.Write(a & Request.QueryString("token")) Response.Write("&CustomData=" & Request.QueryString("CustomData"))
```

## PlayReady Server URL (aka Proxy URL)

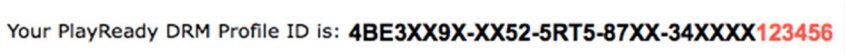
## PX value

The **PX value** is branded for your account and is account specific. Your PX value is always the same no matter what packager you are using.

When generating DRM keys, the EZDRM values returned for **PlayReady LAURL** include the PX number at the end of the Proxy URL (the PX is the last six characters).




It is also the last six characters of your EZDRM Profile ID, as shown in this example:




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

- IP of the end client
- EZDRM TOKEN, this is for EZDRM use only
- PX value, this is your EZDRM specific PX value for your account
- Custom Data is added to the end of the string

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

https://Your_URL.com/auth?IP=10.10.10.10&token=5060ac27-52a7-0151-2bc4-da9055cb0afb&CustomData=samplecustomdata
pX=123456

***Note:*** *There is a line break after Custom Data where the PX value is returned.*

## Custom Data

You can pass custom logic to your authentication URL by adding your own values to the end of your PlayReady LAURL, as follows:

https://playready.ezdrm.com/cency/preauth.aspx?pX=3XXXXX&CustomData=123&CustomData2=&CustomData3=

Depending on your DASH Player, such as the Bitmovin player that has a Custom Data value that uses a Microsoft Custom Data element, you can either use this object or pass custom data via adding values to your PlayReady License Acquisition URL (LA_URL). This is the PlayReady LAURL value that is in returned in your packaging XML license values.

## PlayReady Sample Player

For an online example use the Shaka Player found here:
https://shaka-player-demo.appspot.com/demo/#lang=en-US;build=uncompiled