EZDRM Shaka Packager
DRM Keys Guide
# Table of Contents

**EZDRM Universal DRM**

Shaka Packager – Overview for Raw Key Format 3

**Universal DRM (Widevine & PlayReady)**

- Generating Keys 4
- Universal DRM – Key Value Definitions 7
- Universal DRM – Raw Key Format 7

**Apple FairPlay Streaming**

- Generating Keys 9
- Key Value Definitions 10
- Apple FairPlay Streaming – Key Value Definitions 11
- Apple FairPlay Streaming – Raw Key Format 12

**Additional Information** 13

---

Version 2.1 / Updated June 15, 2021
**Shaka Packager – Overview for Raw Key Format**

This document outlines the raw key format for Shaka Packager and generating DRM keys with EZDRM. For more details visit: https://google.github.io/shaka-packager/html/tutorials/raw_key.html

To download Shaka-Packager visit: https://github.com/google/shaka-packager/releases

**Universal DRM (Widevine & PlayReady)**

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.
Generating Keys

To request the DRM keys from EZDRM to package the media, there are two options, you can call the EZDRM web service in a browser, or you can script this process with curl or other web service calls.

**Option 1: Request DRM keys using EZDRM CPIX Web Service**

1. Call the EZDRM web service in a browser:
   
   ```
   ```

   The parameters are as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>k</td>
<td>kid or Key ID value (client generated) in GUID format*</td>
</tr>
<tr>
<td>u</td>
<td>EZDRM username</td>
</tr>
<tr>
<td>p</td>
<td>EZDRM password</td>
</tr>
<tr>
<td>c</td>
<td>Content ID – generic resource name/identifier (client generated) – passed into id field</td>
</tr>
</tbody>
</table>

   * To generate a GUID for the k value, you can use a GUID generator like the one found here: [http://guid-convert.appspot.com](http://guid-convert.appspot.com).
2. The response from EZDRM will look like this:

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
  <cpix:ContentKeyList>
    <cpix:ContentKey kid="582aXXXX-XXXX-41ef-XXX-ae533ccaXXXX" explicitIV="WCq8DAXXXXXX65TPMpe+A==">
      <cpix:Data>
        <pskc:Secret>
          <pskc:PlainValue>sU+XXXXXqvSN0aXbXXXXX==</pskc:PlainValue>
        </pskc:Secret>
      </cpix:Data>
    </cpix:ContentKey>
  </cpix:ContentKeyList>
  <cpix:DRMSystemList>
    <cpix:DRMSystem kid="582aXXXX-XXXX-41ef-XXX-ae533ccaXXXX" systemId="edefXXXX-79d6-XXXX-a3c8-27dcd5XXXXX"/>
  </cpix:DRMSystemList>
</cpix:CPIX>
```

- **id** – c value returned, generic resource name/identifier (client generated)
- **kid** – Key ID in GUID format (client generated)*
- **pskc:Secret key** – the Secret Content Encryption Key in Base 64 generated by EZDRM and returned as a plain value.

* To generate a GUID for the k value, you can use a GUID generator like the one found here: [http://guid-convert.appspot.com](http://guid-convert.appspot.com).

Here is the example XML return:
Option 2: Request DRM keys with curl

The second option to request DRM keys from EZDRM is to script the process with curl or another web service call.

Using EZDRM's web service, the curl script below retrieves the DRM values from the web service.

```
```
Universal DRM – Key Value Definitions

Widevine

- **key_id**: The kid used for encryption (also known as KID); Base 64 encoded with no dashes.

- **key**: The DRM content encryption key (128 bit key); Base 64 encoded.

For the **key** value use the **pskc:Secret key** value and decode the Plain Value tag from Base 64 to HEX format in lowercase (no dashes). An example decoder can be found at:
https://tomeko.net/online_tools/base64.php?lang=en

```
pskc:Secret key (Base 64) = sU+A8UXXXXvSN0aXXXXwcg==
```

```
(KeyHEX) = b14fXXXX41b77XXXX2374XXXXe2f30XX
```

PlayReady

- **key_id**: This value is the same as the Widevine kid above, used for encryption (also known as KID); Base 64 encoded with no dashes.

- **key**: This value is the same as the Widevine Key above, because the DRM is using common encryption with a shared key base; Base 64 encoded.

Universal DRM – Raw Key Format

The example raw key format for Universal DRM (Widevine & PlayReady):

```
packager-win.exe
in=BigBuckBunny_320x180_Frag.mp4,stream=audio,output=audio.mp4,drm_label=AUDIO
in=BigBuckBunny_320x180_Frag.mp4,stream=video,output=h264_360p.mp4,drm_label=SD
--enable_raw_key_encryption
--keys label=AUDIO: key_id=582aXXXXX41efXXXae533ccaXXX;key=b14fXXX41b77XXX2374XXXe2f30XX, label=SD: key_id=582aXXXXX41efXXXae533ccaXXX;key=b14fXXX41b77XXX2374XXXe2f30XX
--protection_systems Widevine,PlayReady
--mpd_output h264.mpd
```
The example raw key format for Widevine only:

```
packager-win.exe
in=BigBuckBunny_320x180_Frag.mp4,stream=audio,output=audio.mp4,drm_label=AUDIO
in=BigBuckBunny_320x180_Frag.mp4,stream=video,output=h264_360p.mp4,drm_label=SD
--enable_raw_key_encryption
--keys label=AUDIO:key_id=582aXXXXXXXX41efXXXae533ccaXXXX: key=b14fXXXX41b77XXXX2374XXXXe2f30XX, label=SD: key_id=582aXXXXXXXX41efXXXae533ccaXXXX: key=b14fXXXX41b77XXXX2374XXXXe2f30XX
--mpd_output h264.mpd
```

*Note – for Widevine only remove “--protection_systems widevine, PlayReady”*

Output example:

- audio.mp4
- h264.mpd
- h264_360p.mp4
Apple FairPlay Streaming

EZDRM Apple FairPlay DRM is a hosted Apple FairPlay Streaming (DRM). This enables a content owner to encrypt the media with Apple FPS DRM keys and deliver content Apple devices with native support MAC Safari browser via HTML 5 player or iOS via native App or Safari 11.3.

The packaging process encrypts the media. 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.

Generating Keys

Option 1: Request DRM keys using EZDRM CPIX Web Service

3. Call the EZDRM web service in a browser:

   sword&c=resourcename

The parameters are as follows:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>k</strong></td>
<td>kid or Key ID value (client generated) in GUID format*</td>
</tr>
<tr>
<td><strong>u</strong></td>
<td>EZDRM username</td>
</tr>
<tr>
<td><strong>p</strong></td>
<td>EZDRM password</td>
</tr>
<tr>
<td><strong>c</strong></td>
<td>Content ID – generic resource name/identifier (client generated) – passed into id field</td>
</tr>
</tbody>
</table>

* To generate a GUID for the k value, you can use a GUID generator like the one found here: http://guid-convert.appspot.com.
Key Value Definitions

Here are the descriptions of the key values returned by EZDRM:

- **id** – c value returned, generic resource name/identifier (client generated)
- **kid** – Key ID in GUID format (client generated)*
- **pskc:Secret key** – the Secret Content Encryption Key in Base 64 generated by EZDRM and returned as a plain value
- **explicitIV** – the Apple FairPlay explicit IV value

* To generate a GUID for the k value, you can use a GUID generator like the one found here: [http://guid-convert.appspot.com](http://guid-convert.appspot.com).

Here is the example XML return:

```xml
  <cpix:ContentKeyList>
    <cpix:ContentKey kid="582de60c-XXXX-XXXX-a013-XXX33cca5ef8" explicitIV="WCXXXXXXX+gE65TXXeA==">
      <cpix:Data>
        <pskc:Secret>
          <pskc:PlainValue>sU+A8XXXXXXXXXaXbi8wcg==</pskc:PlainValue>
        </pskc:Secret>
      </cpix:Data>
    </cpix:ContentKey>
  </cpix:ContentKeyList>
</cpix:CPIX>
```
**Option 2: Request DRM keys with curl**

The second option to request DRM keys from EZDRM is to script the process with curl or another web service call.

Using EZDRM’s web service, the curl script below retrieves the DRM values from the web service.

```bash
```

**Apple FairPlay Streaming – Key Value Definitions**

Here are the descriptions of the key values returned:

- **key_id**: The kid used for encryption (also known as KID); Base 64 encoded with no dashes (-).
  
- **key**: The DRM content encryption key (128 bit key); Base 64 encoded.

  For the key value use the pskc:Secret key value and decode the Plain Value tag from Base 64 to HEX format in lowercase (no dashes). An example decoder can be found at: [https://tomeko.net/online_tools/base64.php?lang=en](https://tomeko.net/online_tools/base64.php?lang=en)

  `pskc:Secret key (Base 64) = sU+A8UXXXXvSN0aXXXXwcg==`

  
  `(KeyHEX) = b14fXXXX41b77XXXX2374XXXXe2f30XX`

- **iv**: explicitIV decoded from Base 64 to HEX) combined.

  Decode the explicitIV Plain Value Base 64 to HEX format. An example decoder can be found at: [http://guid-convert.appspot.com/](http://guid-convert.appspot.com/)

  `explicitIV (Base 64) = WCq8DXXXX+gE65TXXXX+A==`
iv (HEX no dashes) = 582aXXXXXXXX41efXXXae533ccaXXXX

- **KeyURI** - Use the command line option **--encryption-key-uri** to specify the license URL for encryption. Build by appending the **kid** value to base URL “skd://fps.ezdrm.com/;” for example:
  
  skd://fps.ezdrm.com/;582de60c-XXX-XXX-a013-XXX33cca5ef8

### Apple FairPlay Streaming – Raw Key Format

The example raw key format for Apple HLS:

```bash
packager-win.exe
in=BigBuckBunny_320x180.mp4,stream=audio,output=audio.mp4,drm_label=AUDIO
in=BigBuckBunny_320x180.mp4,stream=video,output=h264_360p.mp4,drm_label=SD
--protection_scheme cbc
--enable_raw_key_encryption
--keys label=AUDIO: key_id=582de60c-XXX-XXX-a013-XXX33cca5ef8: key=b14fXXXX41b77XXXX2374XXXXe2f38XX
label=SD: key_id=582de60c-XXX-XXX-a013-XXX33cca5ef8: key=b14fXXXX41b77XXXX2374XXXXe2f38XX
--protection_systems FairPlay
--iv 582aXXXXXXXX41efXXXae533ccaXXXX
--hls_master_playlist_output h264_master.m3u8
--hls_key_uri skd://fps.ezdrm.com/;582de60c-XXX-XXX-a013-XXX33cca5ef8
```

Output example:
Additional Information

For additional questions and comments please contact: simplify@ezdrm.com