EZDRM Packaging Guide
AES-128 Clear Key with Bento4

Table of Contents

Version 1.0 / Updated September 2020

Introduction

This document outlines generating DRM Keys with EZDRM and utilizing the Bento4 AES Clear Key for packaging for MPEG-DASH and Apple HLS.

Prerequisites

- Bento4 – https://www.bento4.com/downloads/

Generating Keys - DASH

Below are the steps to create the DRM Keys for DASH encryption with Bento4 AES Clear 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 Web Service

1. Call the EZDRM web service in a browser:
   https://cpix.ezdrm.com/clearkey/GenerateKeys.aspx?**k=value**&**r=streamname**&**u=username**&**p=password**

   The parameters are as follows:

Parameter	**Description**
k	kid or Key ID value (client generated) in GUID format*
r	Resource ID (client provided) for resource identification, for example media asset name (also passed back to the authentication URL)
u	EZDRM username
p	EZDRM password

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

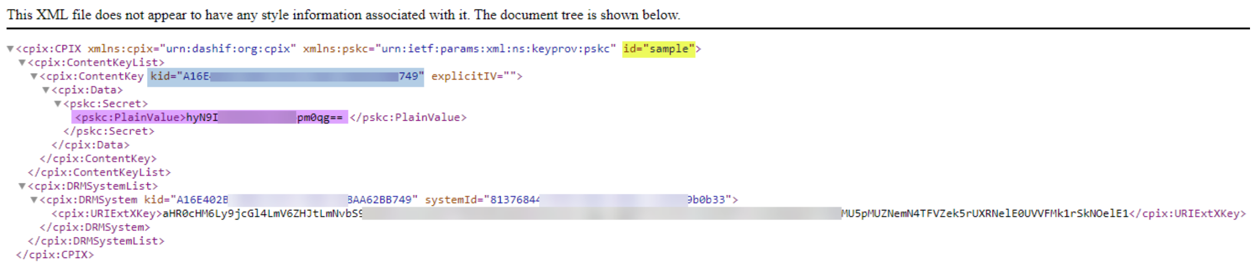
Key Value Definitions

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

- **id** – Resource ID passed back to the authentication URL

- **kid** – Key ID in GUID format

- **pskc:Secret key**– the Secret Content Encryption Key in Base 64 generated by EZDRM and returned as a plain value



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

```
▼<cpix:CPIX xmlns:cpix="urn:dashif:org:cpix" xmlns:pskc="urn:ietf:params:xml:ns:keyprov:pskc" id="sample">
  ▼<cpix:ContentKeyList>
    ▼<cpix:ContentKey kid="A16E[redacted]749" explicitIV="">
      ▼<cpix:Data>
        ▼<pskc:Secret>
           <pskc:PlainValue>hyN9I[redacted]pm0qg== </pskc:PlainValue>
         </pskc:Secret>
       </cpix:Data>
     </cpix:ContentKey>
   </cpix:ContentKeyList>
  ▼<cpix:DRMSystemList>
    ▼<cpix:DRMSystem kid="A16E402B[redacted]BAA62BB749" systemId="81376844[redacted]9b0b33">
       <cpix:URIExtXKey>aHR0cHM6Ly9jcGl4LmV6ZHJtLmNvbS9[redacted]MU5pMUZNemN4TFVZek5rUXRNelE0UVVFMk1rSkNOelE1</cpix:URIExtXKey>
     </cpix:DRMSystem>
   </cpix:DRMSystemList>
</cpix:CPIX>
```

```
<cpix:CPIX xmlns:cpix="urn:dashif:org:cpix" xmlns:pskc="urn:ietf:params:xml:ns:keyprov:pskc" id="sample">
 <cpix:ContentKeyList>
    <cpix:ContentKey kid="A16EXXXX-9056-XXXX-F36D-XXXXXX2BB749" explicitIV="">
     <cpix:Data>
       <pskc:Secret>
           <pskc:PlainValue>hyN9IKXXXXXXXXXXXE5pm0qg== </pskc:PlainValue>
       </pskc:Secret>
     </cpix:Data>
   </cpix:ContentKey>
</cpix:ContentKeyList>
<cpix:DRMSystemList>
   <cpix:DRMSystem kid="A16EXXXX-9056-XXXX-F36D-XXXXXXBB749" systemId="81376844-XXXX-481e-XXXX-cc25dXXXXX3
3">
<cpix:URIExtXKey>aHR0cHM6XXXXXXXXXXXXXXXXXXXXXXGVhcmtleS9ITFNHZXRMaWNlbnNlLmFzcHg/cFg9NTJCMzFDJnI9UVRFM1JUU
XdNa0l0T1RBMU5pMUZNemN4TFVZek5rUXRNelE0UVVFMk1rSkNOelE1</cpix:URIExtXKey>
</cpix:DRMSystem>
</cpix:DRMSystemList>
</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.

```
curl -v https://cpix.ezdrm.com/clearkey/GenerateKeys.aspx?k=value&r=streamname&u=username&p=password
```

Clear Key Packaging - DASH

1. Take the **kid** value from Key Generation and remove the hyphens to create a plain key format:

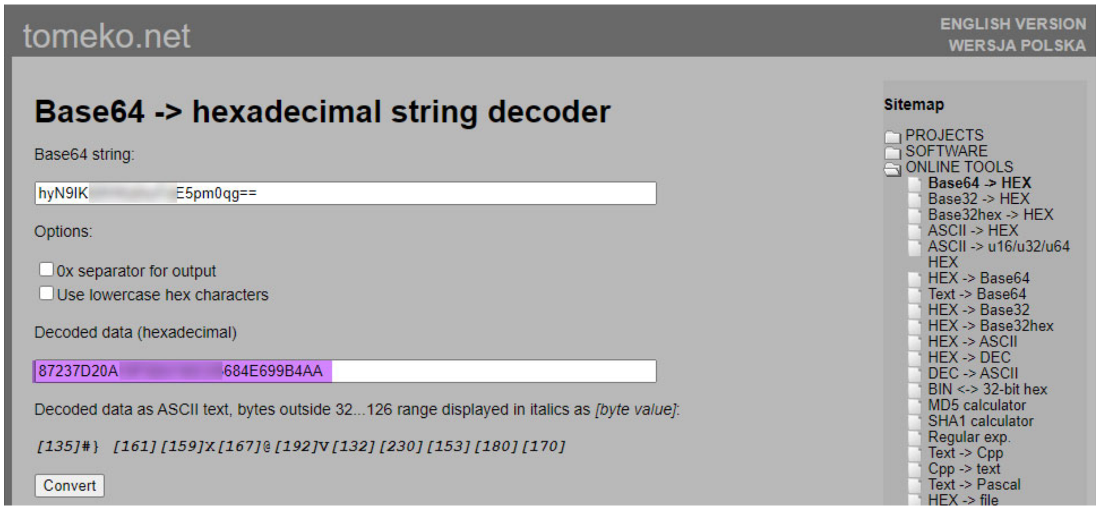
   A16EXXXX-9056-XXXX-F36D-XXXXXX2BB749

   kid = A16EXXXX9056XXXXF36DXXXXXX2BB749

2. Use the **pskc:Secret key** value and decode from Plain Value Base 64 to HEX format. An example decoder can be found at: https://tomeko.net/online_tools/base64.php?lang=en

   (Base 64) hyN9IKXXXXXXXXXXE5pm0qg==

   key (HEX) = 87237D20AXXXXXXXXXXXX05684E699B4AA


tomeko.net

ENGLISH VERSION
WERSJA POLSKA

Base64 -> hexadecimal string decoder

Base64 string:

hyN9IK[redacted]E5pm0qg==

Options:

☐ 0x separator for output
☐ Use lowercase hex characters

Decoded data (hexadecimal)

87237D20A[redacted]684E699B4AA

Decoded data as ASCII text, bytes outside 32...126 range displayed in italics as *[byte value]*:

[135]#} *[161] [159]*x*[167]*@*[192]*v*[132] [230] [153] [180] [170]*

Convert

Sitemap

- PROJECTS
- SOFTWARE
- ONLINE TOOLS
  - **Base64 -> HEX**
  - Base32 -> HEX
  - Base32hex -> HEX
  - ASCII -> HEX
  - ASCII -> u16/u32/u64 HEX
  - HEX -> Base64
  - Text -> Base64
  - HEX -> Base32
  - HEX -> Base32hex
  - HEX -> ASCII
  - HEX -> DEC
  - DEC -> ASCII
  - BIN <-> 32-bit hex
  - MD5 calculator
  - SHA1 calculator
  - Regular exp.
  - Text -> Cpp
  - Cpp -> text
  - Text -> Pascal
  - HEX -> file

3. Take the **kid** value in GUID (with the hyphens) and decode to Base 64, this will become the **laUrl Key** value (the sample encoder can be utilized for this conversion https://tomeko.net/online_tools/hex_to_base64.php?lang=en):

   (GUID) A16EXXXX-9056-XXXX-F36D-XXXXXX2BB749

   laUrl (Base 64) = QTE2RTQwMkItXXXXXXXXXXXXXXNkQtMzQ4QUE2MkJCNzQ5

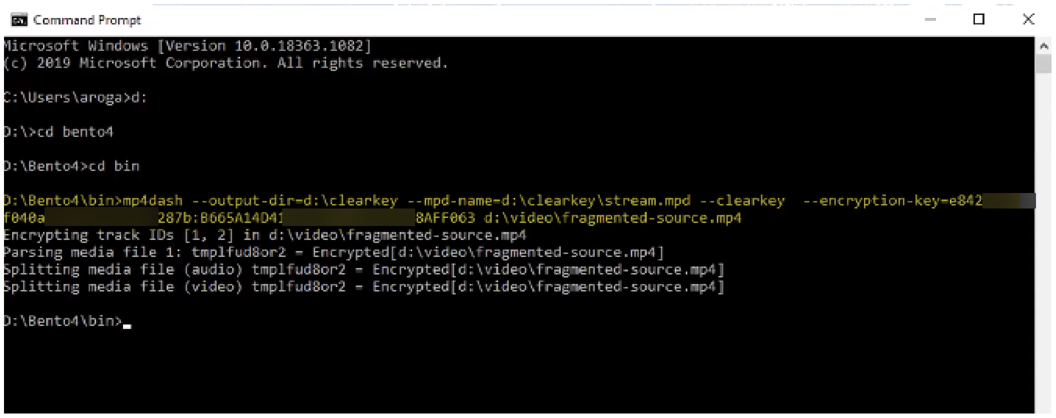
4. Run mp4dash command (included with Bento4) using the [**KID**:**KEY**] format for the encryption key.

```
mp4dash
--output-dir=d:\clearkey\output //the file output directory
--mpd-name=d:\clearkey\stream.mpd //the output .mpd name
--clearkey // DRM type
--encryption-key=A16EXXXX9056XXXXF36DXXXXXX2BB749:87237D20AXXXXXXXXXXXX05684E699B4AA //encryption key
d:\video\fragmented-source.mp4 //the source file location
```

Note: This value is not currently required:

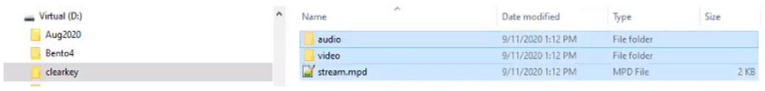
```
--clearkey-license-uri=<<LA_URL>>
```

Here is an example of the mp4dash command:



```
Command Prompt
Microsoft Windows [Version 10.0.18363.1082]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\aroga>d:

D:\>cd bento4

D:\Bento4>cd bin

D:\Bento4\bin>mp4dash --output-dir=d:\clearkey --mpd-name=d:\clearkey\stream.mpd --clearkey  --encryption-key=e842[redacted]
f040a[redacted]287b:B665A14D41[redacted]8AFF063 d:\video\fragmented-source.mp4
Encrypting track IDs [1, 2] in d:\video\fragmented-source.mp4
Parsing media file 1: tmplfud8or2 = Encrypted[d:\video\fragmented-source.mp4]
Splitting media file (audio) tmplfud8or2 = Encrypted[d:\video\fragmented-source.mp4]
Splitting media file (video) tmplfud8or2 = Encrypted[d:\video\fragmented-source.mp4]

D:\Bento4\bin>_
```

5. Upload the contents created in the Output directory to your file server. The URL to the .mpd will become your **manifest URL**.



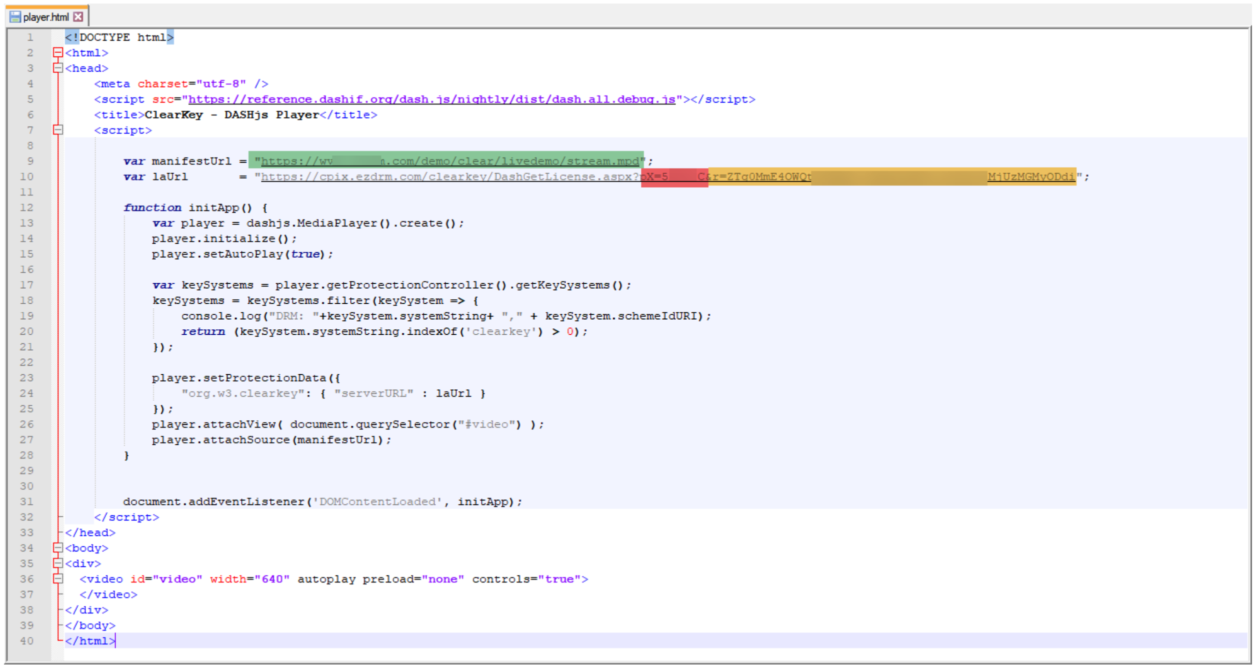
Name	Date modified	Type	Size
audio	9/11/2020 1:12 PM	File folder	
video	9/11/2020 1:12 PM	File folder	
stream.mpd	9/11/2020 1:12 PM	MPD File	2 KB

Testing Playback - DASH

For Clear Key playback, download the .zip for this html file from EZDRM: https://wvm.ezdrm.com/demo/clear/sample04/player.html

Clear Key can utilize any browser such as Chrome, IE, FireFox, etc.

1. Modify the **player.html** with the **manifest URL** (created in Step 5) and the **var laUrl** base url values.



```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <script src="https://reference.dashif.org/dash.js/nightly/dist/dash.all.debug.js"></script>
    <title>ClearKey - DASHjs Player</title>
    <script>

        var manifestUrl = "https://wv[redacted]n.com/demo/clear/livedemo/stream.mpd";
        var laUrl       = "https://cpix.ezdrm.com/clearkey/DashGetLicense.aspx?pX=5[redacted]C&r=ZTg0MmE4OWQt[redacted]MjUzMGMyODdi";

        function initApp() {
            var player = dashjs.MediaPlayer().create();
            player.initialize();
            player.setAutoPlay(true);

            var keySystems = player.getProtectionController().getKeySystems();
            keySystems = keySystems.filter(keySystem => {
                console.log("DRM: "+keySystem.systemString+ "," + keySystem.schemeIdURI);
                return (keySystem.systemString.indexOf('clearkey') > 0);
            });

            player.setProtectionData({
                "org.w3.clearkey": { "serverURL" : laUrl }
            });
            player.attachView( document.querySelector("#video") );
            player.attachSource(manifestUrl);
        }


        document.addEventListener('DOMContentLoaded', initApp);
    </script>
</head>
<body>
<div>
  <video id="video" width="640" autoplay preload="none" controls="true">
  </video>
</div>
</body>
</html>
```

- The **var laUrl** is the base URL plus the **PX Value** and the **r value** is the **laUrl key** in Base64.

  https://cpix.ezdrm.com/clearkey/DashGetLicense.aspx?**pX=XXXXXX**&**r=QTE2RTQwMkItXXXXXXXXXXXXXXNkQtMzQ4QUE2MkJCNzQ5**

Notes:

- The Clear Key MPEG-DASH **PX Value** is provided by EZDRM.

- Append to the end of the **laUrl** any **Custom Data**. The **Custom Data** is sent back to your Authorization URL for additional business logic and validation.

  https://cpix.ezdrm.com/clearkey/DashGetLicense.aspx?pX=XXXXXX&r=QTE2RTQwMkItXXXXXXXXXXXXXXNkQtMzQ4QUE2MkJCNzQ5**&customdata=123**

```
<html>
<head>
    <meta charset="utf-8" />
          <script src="https://reference.dashif.org/dash.js/nightly/dist/dash.all.debug.js"></script>
    <title>ClearKey - DASHjs Player</title>
    <script>

          var manifestUrl = "https://website.com/demo/clear/sample04/stream.mpd";
          var laUrl       = "https://cpix.ezdrm.com/clearkey/DashGetLicense.aspx?pX=XXXXXX&r=QTE2RTQwMkIt
XXXXXXXXXXXXXXNkOtMzO4OUE2MkJCNzO5";
        function initApp() {
            var player = dashjs.MediaPlayer().create();
            player.initialize();
                               player.setAutoPlay(true);

                               var keySystems = player.getProtectionController().getKeySystems();
                               keySystems = keySystems.filter(keySystem => {
                                          console.log("DRM: "+keySystem.systemString+ "," + keySystem.sche
meIdURI);
                                          return (keySystem.systemString.indexOf('clearkey') > 0);
                               });

                               player.setProtectionData({
                                          "org.w3.clearkey": { "serverURL" : laUrl }
                               });
                               player.attachView( document.querySelector("#video") );

                               player.attachSource(manifestUrl);
                     }


                     document.addEventListener('DOMContentLoaded', initApp);
    </script>
</head>
<body>
<div>
  <video id="video" width="640" autoplay preload="none" controls="true">
  </video>
</div>
</body>
</html>
```

2. Open the **player.html** in a browser for testing playback.

Generating Keys - HLS

Below are the steps to create the DRM Keys for Apple HLS encryption with Bento4 AES Clear 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 Web Service

1. Call the EZDRM web service in a browser:
   https://cpix.ezdrm.com/clearkey/GenerateKeys.aspx?**k=value**&**r=streamname**&**u=username**&**p=password**

   The parameters are as follows:

Parameter	Description
k	kid or Key ID value (client generated) in GUID format*
r	Resource ID (client provided) for resource identification, for example media asset name (also passed back to the authentication URL)
u	EZDRM username
p	EZDRM password

* 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

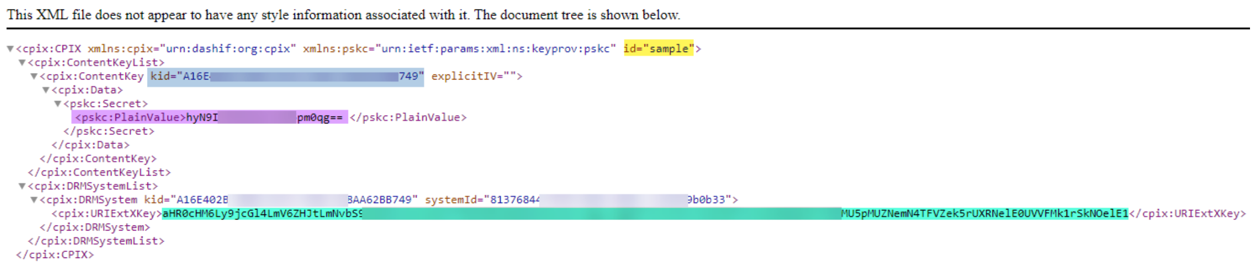
Here are the descriptions of the key values returned by EZDRM:

- **id** – Resource ID passed back to the authentication URL

- **kid** – Key ID in GUID format

- **pskc:Secret key**– the Secret Content Encryption Key in Base 64 generated by EZDRM and returned as a plain value

- **URIExtXKey** – the Apple FairPlay License URL for encryption



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

```
▼<cpix:CPIX xmlns:cpix="urn:dashif:org:cpix" xmlns:pskc="urn:ietf:params:xml:ns:keyprov:pskc" id="sample">
  ▼<cpix:ContentKeyList>
    ▼<cpix:ContentKey kid="A16E[redacted]749" explicitIV="">
      ▼<cpix:Data>
        ▼<pskc:Secret>
           <pskc:PlainValue>hyN9I[redacted]pm0qg== </pskc:PlainValue>
         </pskc:Secret>
       </cpix:Data>
     </cpix:ContentKey>
   </cpix:ContentKeyList>
  ▼<cpix:DRMSystemList>
    ▼<cpix:DRMSystem kid="A16E402B[redacted]BAA62BB749" systemId="81376844[redacted]9b0b33">
       <cpix:URIExtXKey>aHR0cHM6Ly9jcGl4LmV6ZHJtLmNvbS[redacted]MU5pMUZNemN4TFVZek5rUXRNelE0UVVFMk1rSkNOelE1</cpix:URIExtXKey>
     </cpix:DRMSystem>
   </cpix:DRMSystemList>
 </cpix:CPIX>
```

```
<cpix:CPIX xmlns:cpix="urn:dashif:org:cpix" xmlns:pskc="urn:ietf:params:xml:ns:keyprov:pskc" id="sample">
 <cpix:ContentKeyList>
    <cpix:ContentKey kid="A16EXXXX-9056-XXXX-F36D-XXXXXX2BB749" explicitIV="">
     <cpix:Data>
       <pskc:Secret>
           <pskc:PlainValue>hyN9IKXXXXXXXXXXXE5pm0qg== </pskc:PlainValue>
       </pskc:Secret>
     </cpix:Data>
   </cpix:ContentKey>
</cpix:ContentKeyList>
<cpix:DRMSystemList>
   <cpix:DRMSystem kid="A16EXXXX-9056-XXXX-F36D-XXXXXXBB749" systemId="81376844-XXXX-481e-XXXX-cc25dXXXXX3
3">
<cpix:URIExtXKey>aHR0cHM6XXXXXXXXXXXXXXXXXXXXXXGVhcmtleS9ITFNHZXRMaWNlbnNlLmFzcHg/cFg9NTJCMzFDJnI9UVRFMlJUU
XdNa0l0T1RBMU5pMUZNemN4TFVZek5rUXRNelE0UVVFMk1rSkNOelE1</cpix:URIExtXKey>
</cpix:DRMSystem>
</cpix:DRMSystemList>
</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.

```
curl -v https://cpix.ezdrm.com/clearkey/GenerateKeys.aspx?k=value&r=streamname&u=username&p=password
```

Clear Key Packaging - Apple HLS

1. Use the **pskc:Secret key** value and decode from Plain Value Base 64 to HEX format. An example decoder can be found at: https://tomeko.net/online_tools/base64.php?lang=en

   (Base 64) hyN9IKXXXXXXXXXXXE5pm0qg==

   key (HEX) = 87237D20AXXXXXXXXXXXXX05684E699B4AA

1. Use the **URIExtXKey** from Key generation and decode from Base 64 for the **Encryption Key URI** value.

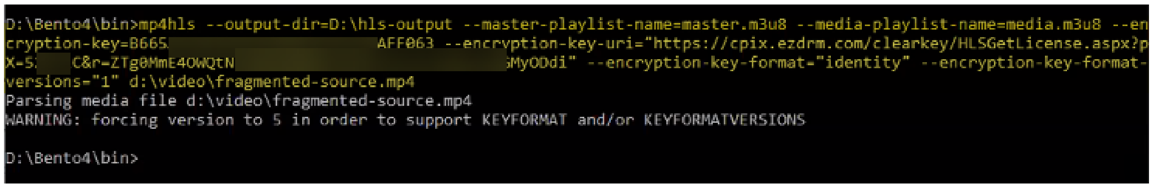
   (Base 66) = aHR0cHM6XXXXXXXXXXXXXXXXXXXXXGVhcmtleS9ITFNHZXRMaWNlbnNlLmFzcHg/cFg9NTJCMzFDJnI9UVRFMlJUUXdNa0l0T1RBMU5pMUZNemN4TFVZek5rUXRNelE0UVVFMk1rSkNOelE1

   https://cpix.ezdrm.com/clearkey/HLSGetLicense.aspx?pX=XXXXX&r=QTE2RTQwMkItXXXXXXXXXXXXXXNkQtMzQ4QUE2MkJCNzQ5

2. Run mp4hls command (included with Bento4):

```
mp4hls
--output-dir=D:\hls-output //the file output directory
--master-playlist-name=master.m3u8 //playlist name
--media-playlist-name=media.m3u8  // media playlist name
--encryption-key=87237D20AXXXXXXXXXXXXX05684E699B4AA // Secret Key in HEX format
--encryption-key-uri="https://cpix.ezdrm.com/clearkey/DashGetLicense.aspx?pX=XXXXXX&r=QTE2RTQwMkIt
XXXXXXXXXXXXXXNkQtMzQ4QUE2MkJCNzQ5" // Decoded Encryption Key URI
--encryption-key-format="identity" // Key encryption format (default)
--encryption-key-format-versions="1" // Key encryption format version (default)
d:\video\fragmented-source.mp4 //the source file location
```

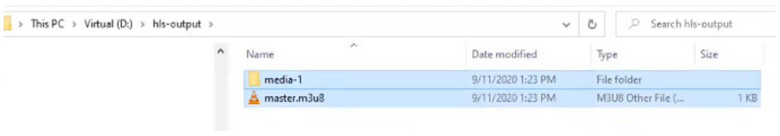
Here is an example of the mp4hls command:



```
D:\Bento4\bin>mp4hls --output-dir=D:\hls-output --master-playlist-name=master.m3u8 --media-playlist-name=media.m3u8 --en
cryption-key=B665[redacted]AFF063 --encryption-key-uri="https://cpix.ezdrm.com/clearkey/HLSGetLicense.aspx?p
X-5[redacted]C&r=ZTg0MmE4OWQtN[redacted]MyODdi" --encryption-key-format="identity" --encryption-key-format-
versions="1" d:\video\fragmented-source.mp4
Parsing media file d:\video\fragmented-source.mp4
WARNING: forcing version to 5 in order to support KEYFORMAT and/or KEYFORMATVERSIONS

D:\Bento4\bin>
```

Testing Playback - HLS

1. Upload the contents created in the Output directory to your file server. The URL to the .m3u8 will become your **manifest URL**.



Name	Date modified	Type	Size
media-1	9/11/2020 1:23 PM	File folder	
master.m3u8	9/11/2020 1:23 PM	M3U8 Other File (...	1 KB

2. For testing playback, use a demo player such as: http://viblast.com/player/demo-user-stream/ and enter the Stream URL.

Additional Information

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