EZDRM
Bitmovin Player Integration

Table of Contents

Version 1.0 / Updated August 14, 2020

Introduction

This document outlines the steps necessary to successfully integrate EZDRM with the Bitmovin player.

Bitmovin accepts streams from an MPEG-DASH (*.mpd) file for Microsoft PlayReady and Google Widevine, and/or a *.m3u8 file for Apple FairPlay and HLS.

For information about the Bitmovin web video player, see:

- Bitmovin player homepage: https://Bitmovin.com/
- Bitmovin player online guide: https://bitmovin.com/docs/player

Requirements

1. Basic understanding of HLS, MPEG/DASH, and other streaming protocols and formats.
2. Bitmovin Player HTML Plug-in Code.

Preparation in Bitmovin

Log in to your Bitmovin account (bitmovin.com) and perform these steps:

1. Using left side navigation bar, click **Player → Licenses → default-licenses**
2. Copy your "Player Key" to a convenient place for use later.
3. Add your domain to the "Domains/ Package Name / Bundle Identifiers" list.

Sample Code Files

EZDRM provides complete sample html files that you can configure and use, based on those from Bitmovin.com:

- dash-sample.html [Widevine and PlayReady]
- fairplay-sample.html
- combined-sample.html [Widevine, PlayReady, and Fairplay]

You can download these files at:

https:// www.ezdrm.com/downloads/Players/BitMovin/BitMovin-EZDRM-sample.zip

The following sections show how to customize the html files.

Head Section Code

```
<! --  Base Bitmovin player code -->
<script type="text/javascript" src="https://cdn.bitmovin.com/player/web/8.1.0/bitmovinplayer.js"></script>
```

Widevine/PlayReady Universal DRM

MPEG-DASH requires 3 values:

1. The **.mpd URL**
2. The **Widevine Proxy URL**
3. The **PlayReady Proxy URL**

Obtaining a PX Value

You will need your six-digit PXs values as part of the LA_URLs in the upcoming sections.

Your Widevine PX value is the last six characters of your Widevine Profile ID. Your PlayReady PX value is the last six characters of your PlayReady Profile ID. The appropriate one is required for all packagers you use.

Visit https://www.ezdrm.com/Documentation/EZDRM_Testing_Playback_v2.pdf for more details on how to find your PX value.

LA_ URL - Widevine

For all packagers (except AWS, Anevia, and related ones), the base LA_URL is:
https://widevine-dash.ezdrm.com/proxy?pX=XXXXXX

Replace **XXXXXX** with your Widevine PX Value.

Widevine for AWS, Anevia, and other CPIX or SPEKE-based encoders.

The Widevine LA_URL is different for these packagers. The base URL is:
https://widevine-dash.ezdrm.com/widevine-php/widevine-foreignkey.php?pX= XXXXXX

Replace **XXXXXX** with your Widevine PX Value.

LA_ URL - PlayReady

For PlayReady, the base proxy URL is:
https://playready.ezdrm.com/cency/preauth.aspx?pX= XXXXXX

Replace **XXXXXX** with your PlayReady PX Value.

Sample Widevine/PlayReady Script

```
<!--  Start the player   -->
<script type="text/javascript">
     const container = document.getElementById('my-player');
     const playerConfig = {
         key: '<<YOUR PLAYER KEY>>' <!-- Replace with your URL-->
            drm: {
     };
     const source = {
         dash: '<<HTTPS LOCATION OF YOUR MPD FILE>>',  <!-- Replace with your URL-->
            drm: {
         widevine: {
                   <!-- Replace “XXXXXX” with your Widevine PX Value -->
          LA_URL: 'https://widevine-dash.ezdrm.com/widevine-php/widevine-foreignkey.php?pX=XXXXXX'
         },
         playready: {
                    <!-- Replace “XXXXXX” with your PlayReady PX Value -->
          LA_URL: 'https://playready.ezdrm.com/cency/preauth.aspx?pX=XXXXXX'
         }
            }
          };
          <!--  Bitmovin Specific required player code   -->
          const player = new bitmovin.player.Player(container, playerConfig);
          player.load(source).then(
         player => {
             console.log('Successfully created Bitmovin Player instance')
         },
         reason => {
             console.log('Error while creating Bitmovin Player instance')
         }
     );
     </script>
```

Apple FairPlay DRM for HLS

Apple FairPlay requires 3 values.

1. The **.m3u8 URL**
2. The **license_url**
   - The base license_url is **https://fps.ezdrm.com/api/licenses/<<ASSET-ID>>**
3. The **FPS cert URL**
   - This is the .cer file provided by Apple. This URL should be either on your Website or CDN.

Sample Fairplay Script

```
<script type="text/javascript">
     const container = document.getElementById('my-player');
     const playerConfig = {
         key: '<<YOUR PLAYER KEY>>' <!-- Replace with your URL-->
     };
     const source = {
         hls: '<<HTTPS LOCATION OF YOUR M3U8 FILE>>',  <!-- Replace with your URL-->
          drm: {
          fairplay: {
             LA_URL: 'https://fps.ezdrm.com/api/licenses/<<ASSET-ID>>',  <!-- Insert Asset ID -->
             certificate_url: '<<HTTPS LOCATION OF YOUR CERTIFICATE>>',  <!-- Replace with your URL-->
          <!--  Additional required code   -->
  prepareContentId: function(contentId) {
         var uri = contentId;
         var uriParts = uri.split('://', 1);
         var protocol = uriParts[0].slice(-3);
         uriParts = uri.split(';', 2);
         contentId = uriParts.length>1?uriParts[1]:'';
         return protocol.toLowerCase()=='skd'?contentId:'';
       },
       prepareLicenseAsync: function(ckc) {
         return new Promise(function(resolve, reject) {
           var reader = new FileReader();
           reader.addEventListener('loadend', function() {
             var array = new Uint8Array(reader.result);
             resolve(array);
           });
           reader.addEventListener('error', function() {
             reject(reader.error);
           });
```

```
            reader.readAsArrayBuffer(ckc);
          });
        },
        prepareMessage: function(event, session) {
          return new Blob([event.message], {type: 'application/octet-binary'});
        },
        headers: [{
          name: 'content-type',
          value: 'application/octet-stream'
        }],
        useUint16InitData: true,
        licenseResponseType: 'blob'
        }
  }
};
const player = new bitmovin.player.Player(container, playerConfig);
player.load(source).then(
    player => {
        console.log('Successfully created Bitmovin Player instance')
    },
    reason => {
        console.log('Error while creating Bitmovin Player instance')
    }
);
</script>
```

Sample Combined Script

For Widevine, PlayReady, and Apple Fairplay

```
<script type="text/javascript">
     const container = document.getElementById('my-player');
     const playerConfig = {
         key: '<<YOUR PLAYER KEY>>' <!-- Replace with your URL-->'
     };
     const source = {
         dash: '<<HTTPS LOCATION OF YOUR MPD FILE>>',  <!-- Replace with your URL-->
          hls: '<<HTTPS LOCATION OF YOUR M3U8 FILE>>',  <!-- Replace with your URL-->
```

```
        drm: {
      widevine: {
                <!-- Replace “XXXXXX” with your Widevine PX Value -->
       LA_URL: 'https://widevine-dash.ezdrm.com/widevine-php/widevine-foreignkey.php?pX=XXXXXX'
      },
      playready: {
                 <!-- Replace “XXXXXX” with your PlayReady PX Value -->
       LA_URL: 'https://playready.ezdrm.com/cency/preauth.aspx?pX=XXXXXX'
      },
       fairplay: {
        LA_URL: 'https://fps.ezdrm.com/api/licenses/<<ASSET-ID>>',  <!-- Insert Asset ID -->
        certificateURL: '<<HTTPS LOCATION OF YOUR CERTIFICATE>>',  <!-- Replace with your URL-->
       <!--  Bitmovin Specific required player code   -->
    prepareContentId: function(contentId) {
      var uri = contentId;
      var uriParts = uri.split('://', 1);
      var protocol = uriParts[0].slice(-3);
      uriParts = uri.split(';', 2);
      contentId = uriParts.length>1?uriParts[1]:'';
      return protocol.toLowerCase()=='skd'?contentId:'';
    },
    prepareLicenseAsync: function(ckc) {
      return new Promise(function(resolve, reject) {
        var reader = new FileReader();
        reader.addEventListener('loadend', function() {
          var array = new Uint8Array(reader.result);
          resolve(array);
        });
        reader.addEventListener('error', function() {
          reject(reader.error);
        });
        reader.readAsArrayBuffer(ckc);
      });
    },
    prepareMessage: function(event, session) {
      return new Blob([event.message], {type: 'application/octet-binary'});
    },
    headers: [{
      name: 'content-type',
      value: 'application/octet-stream'
    }],
    useUint16InitData: true,
    licenseResponseType: 'blob'
          }
        }
```

```
        };
        const player = new bitmovin.player.Player(container, playerConfig);
        player.load(source).then(
        player => {
            console.log('Successfully created Bitmovin Player instance')
        },
        reason => {
            console.log('Error while creating Bitmovin Player instance')
        }
    );
</script>
```

Additional Information

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