



Videon EdgeCaster and NexPlayer Low Latency HTTP Streaming & Compatibility Test Report

Project Name	Low Latency HTTP Streaming and Compatibility Test
Project Purpose	The purpose of this project is to validate and measure the end to end latency between Videon's EdgeCaster encoder and NexPlayer's player application in conjunction with AWS MediaStore in low latency, CMAF, and MBR workflows.

Test Highlights

Test Purpose: Validate and measure the latency using the NexPlayer player application on different platforms when video is encoded by Videon's EdgeCaster and sent to NexPlayer via AWS. Validation and measurements will be examined in low latency modes using CMAF along with Videon's multiple bit rate support.

Test Goal: The goal is to clearly identify end to end workflow compatibility along with measuring the expected latency for different workflows. Quantitative measurements ensure both companies have a baseline reference for on-going engineering improvement. The results can also be used to showcase recommended workflows and expected results to interested parties. In performing the tests, attention will be paid to device synchronization.

Test Results: Videon has established a goal of achieving under 3 seconds end to end latency as our basis for recommending a low latency, HTTP workflow. Using either a DASH or HLS low latency workflow with CMAF chunking, an end to end latency of three seconds or less was measured using Android and iOS playback. Using a DASH low latency workflow with CMAF chunking, an end to end latency of less than three seconds was measured using HTML5 playback. Measurement of HLS low latency CMAF chunking on HTML5 did not produce workflows that enable lower than 3 seconds of latency at this time, thus HTML5 cannot be recommended for HLS ULL workflows. Synchronization across devices was only found possible using Android and iOS, thus HTML cannot be recommended since HTML synchronization was not within 1 second when enabled. In all instances, DASH/HLS in non-low latency mode is fully supported.

Intended Audience: This testing document assumes the ability to set up or access an AWS account as well as have access to the NexPlayer player applications. With this knowledge, one should be able to use this information to successfully run all tests within 2 hours.



Test Purpose

Videon's goal in the following testing is to validate and quantify the glass to glass latency for the workflow involving Videon's products, AWS as a cloud service provider (in this case HTTP Origin and CDN), and NexPlayer's player client. Understanding this workflow and its associated latency ensures an understanding of the performance between NexPlayer and Videon. This will position Videon and its partners with the ability to quickly and easily recommend the verified workflow to customers and partners.

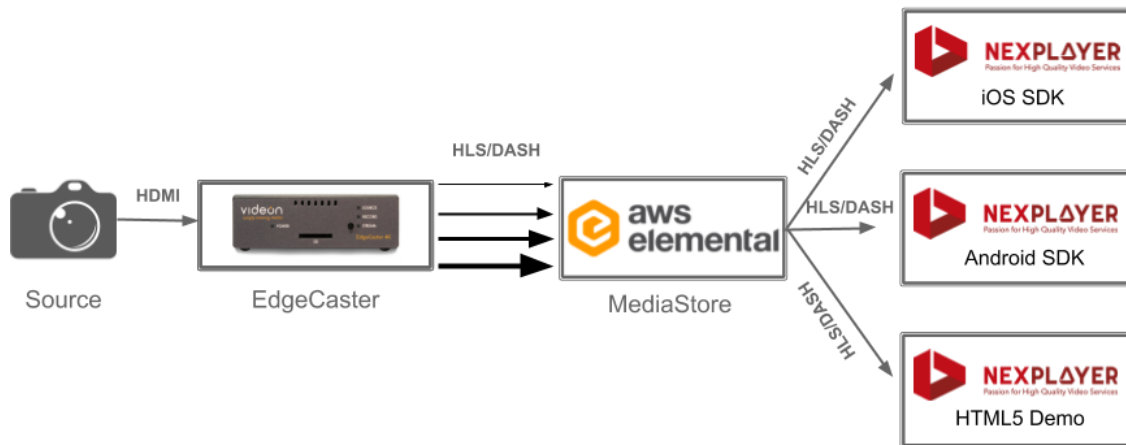
Test Goal

The following testing aims to measure end to end latency with several different workflows. Initial tests will be done using a single bit rate, non CMAF configuration. A second base measurement will be done using HLS/DASH with Videon's multiple bit rate capability. Low latency modes will be measured using a single bit stream with CMAF enabled. A second low latency test will be measured using Videon's multiple bitrate stream with CMAF enabled. As applicable to the workflow, NexPlayer's player client will be tested on iOS and Android platforms along with a HTML5 browser. Both qualitative and quantitative measurements will be recorded.

Test Details

Product Used	Hardware	Version Number
Videon EdgeCaster		0.3.1067.61.3 (MBR V0.2)
NexPlayer Sample App (iOS)	Apple iPad 5th Gen (Model # MP2F2LL/A, Software Version 13.3.1 (17D50))	5.40.1.5136
NexPlayer APK (Android)	Android OnePlus (Model # A3000, Android Version 6.0.1) Motorola moto g6 (Model # XT1925-6, Android Version 9)	NexPlayerSDK_Android_WV_MediaDRM_ver6.69.2.815_20200217_HLS_DASH_PD_Local_VAST_NativeAudio_APPID_by20200517.apk
NexPlayer Demo Player (HTML5)	Dell Latitude (Ubuntu 18.04) Apple Macintosh Dell Latitude (Windows 10, Version 1909)	Release_3.4-531-gd2f1a963-dirty-development
AWS MediaStore	N/A	N/A

Block Diagram



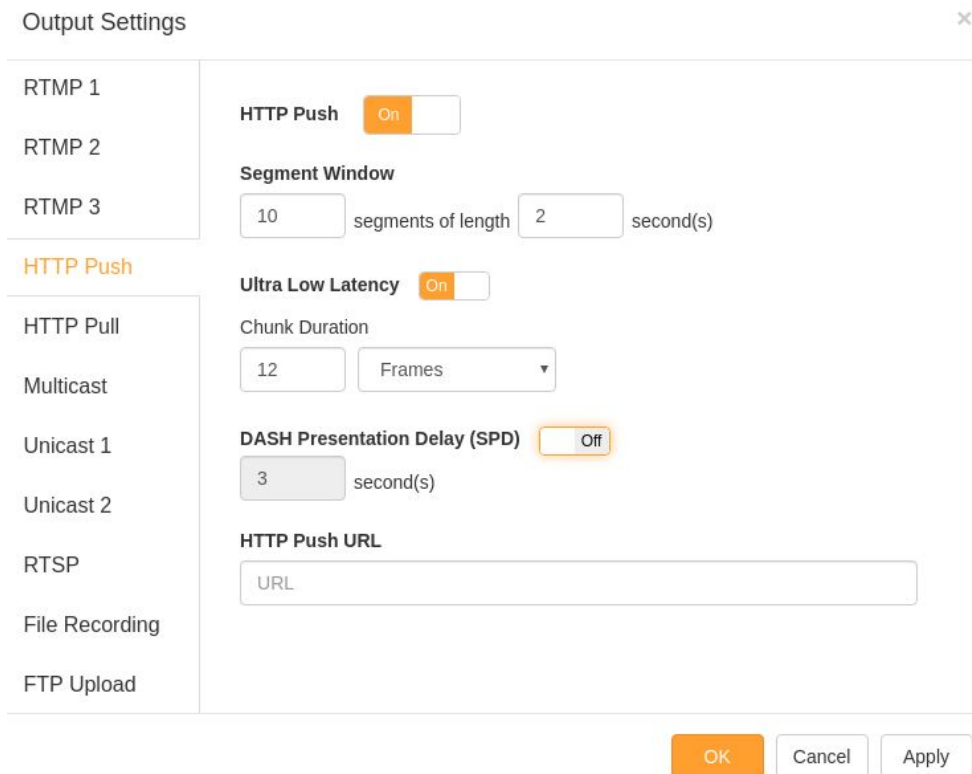
Test Instructions

To configure AWS MediaStore

1. Open the CloudFormation template by clicking here: [ULL CloudFormation Template](#)
2. Select the region closest to you from the title bar in the upper right corner of the CloudFormation screen, options are N. Virginia, Oregon, Seoul, Sydney, Tokyo, Frankfurt, Ireland, and Stockholm
3. From the “Create Stack” screen, select the “Next” button on the bottom right of the screen
4. From the “Specify stack details”, enter the following fields:
 - a. **Stack Name** – The default is “ULL”, you can name it something different, but it cannot be the same name as an existing stack
 - b. **MediaStoreContainerName** – The default is “MyContainer”, you can name it something different, but no spaces, just numbers/characters
 - c. **EncoderIP** – This is the encoder’s public IP
 - i. If you want to allow traffic from anywhere, enter 0.0.0.0
 - d. **EnableCloudFront** – The default is “False”. Set this to true if you want to configure a CloudFront distribution to serve your content. This is recommended, but it adds about 15 minutes to the orchestration time for this template.
5. Leave all the other fields unmodified, and select the “Next” button at the bottom right of the screen
6. From the “Configure stack options” screen select the “Next” button at the bottom right of the screen
7. From the “Review Stackname” screen, check the Capabilities acknowledgment and select the “Create stack” button at the bottom right of the screen.
8. When the stack is complete, you should see a green “CREATE_COMPUTE” check box under the Stacks column. When deploying with CloudFront set to true, this should take 20 minutes. When deploying without CloudFront, it should take less than 3 minutes.

To configure EdgeCaster as **Single Bitrate**

1. Open **Encoder Settings** in the Videon EdgeCaster web UI and set the **Quality/Latency setting to Lowest**.
2. **Navigate to the HTTP Push tab within Output Settings** of the Videon EdgeCaster



The screenshot shows the 'Output Settings' dialog box with the 'HTTP Push' tab selected. The settings are as follows:

- RTMP 1**: HTTP Push On
- RTMP 2**: Segment Window: 10 segments of length 2 second(s)
- RTMP 3**: Ultra Low Latency On
- HTTP Push**:
 - Chunk Duration: 12 Frames
 - DASH Presentation Delay (SPD) Off
 - 3 second(s)
 - HTTP Push URL:
- HTTP Pull**
- Multicast**
- Unicast 1**
- Unicast 2**
- RTSP**
- File Recording**
- FTP Upload**

Buttons at the bottom: OK, Cancel, Apply

3. Enter the **ingest URL** for the desired origin server into the **HTTP Push URL** field
4. Configure the following settings
 - o 5 segments of length 6 seconds
 - o TS for TS tests, fMP4 CMAF tests
 - o ULL ON
 - 6-frame chunks
 - o DASH SPD ON
 - 2 seconds
5. Turn **HTTP Push** ON and click **OK** or **Apply**
6. You can now view the stream at **[HTTP Push URL]/master.m3u8** or **[HTTP Push URL]/manifest.mpd**

To configure EdgeCaster as **Multiple Bitrate**

1. Click **+ Add MBR Group**
2. Click **Group Settings** (it appears in place of + Add MBR Group)
3. Set **Quality/Latency = Lowest**
4. Click **Save**
5. Click **+ Add Video Profile** for the number of Video Profiles in the ladder
 - Do not exceed 4Kp30 for combined resolution/framerate
 - For this testing, the following ladder was used
 - 1080p60 @ 5000kbps
 - 720p60 @ 2000kbps
 - 480p60 @ 1000kbps
 - 360p60 @ 750kbps
6. For each **Video Profile**, click the name in the left sidebar then
 - Click the name in the main configuration menu (should have a **pencil icon** next to it) and type the new name of the Video Profile
 - Set **Video Scaling** to the desired resolution for the profile
 - Configure the rest of the settings according to the desired ladder
 - Make sure to turn **MBR Group Member** to **ON**
 - Click **Save**

AV Input

Input Settings

Video Profiles

1080 @ 5mbps

720 @ 2mbps

480 @ 1mbps

360 @ .75mbps

+ Add Profile

MBR Group

Group Settings

Audio Profiles

+ Add Profile

Outputs

HTTP Pull

HTTP Push

1080 @ 5mbps ✎

Video Input Resolution: 1080p60

Video Scaling

1920x1080p

Limit to 30 FPS Off

Encoded Video Resolution: 1080p60

Encoding Mode

Variable Bitrate

Video Bitrate

5000 kbps

Video Encoding

H.264 (AVC)

H.264 Profile

High Profile

Keyframe Interval

1 Second(s)

Quality/Latency

Normal

MBR Group Member On

Settings Saved.

Save

7. Under **Audio Profiles** click **+ Add Audio Profile**
8. Click the new **Audio Profile name** that appears
9. Click the name in the main configuration menu (should have a **pencil icon** next to it) and type the new name of the Audio Profile
10. Configure the desired settings
11. Click **Save**
12. In **HTTP Push**, select the **MBR Group** as the **Video Source**
13. Select the previously created **Audio Profile** and the **Audio Source**
14. Enter the **ingest URL** for the desired origin server into the **HTTP Push URL** field
15. Configure the following settings
 - 5 segments of length 6 seconds
 - TS for TS tests fMP4 CMAF tests
 - ULL ON
 - 6-frame chunks
 - DASH SPD ON
 - 2 seconds
16. Turn **HTTP Push** ON and click **OK** or **Apply**
17. You can now view the stream at **[HTTP Push URL]/master.m3u8** or **[HTTP Push URL]/manifest.mpd**

AV Input

Input Settings

Video Profiles

1080 @ 5mbps

720 @ 2mbps

480 @ 1mbps

360 @ .75mbps

+ Add Profile

MBR Group

Group Settings

Audio Profiles

AAC @ 128kbps

+ Add Profile

Outputs

HTTP Pull

HTTP Push

HTTP Push On

Video Source

MBR Group

Audio Source

AAC @ 128kbps

Segment Window

5 segments of length 6 second(s)

Media Type

fMP4

Ultra Low Latency On

Chunk Duration

6 Frames

DASH Presentation Delay (SPD) Off

2 second(s)

HTTP Push URL

https://myMediaStoreURL.com/testing

Save

To configure NexPlayer on **Android**:

1. From the main screen, tap the three vertical dots in the top right corner and tap **settings**
2. Configure the following settings
 - a. Low Latency
 - i. Enabled = True
 - ii. Low Latency Buffer Option = Auto Buffer Management
 - b. SPD Settings
 - i. Enabled = True
 - ii. SPD delay time = 3000ms
 - iii. SPD sync value = 500ms
 - iv. SPD too much sync diff value = 10000ms
3. Navigate back to the main screen
4. Tap streaming at the top
5. Tap **GO TO URL**
6. In the **URL** field, enter the playback URL in the last step of the [EdgeCaster Configuration](#)
7. Tap **PLAY** at the bottom to start the stream

To configure NexPlayer on **iOS**:

1. From the main screen, tap the **gear** in the bottom right corner
2. Scroll through the settings and configure the following
 - a. Low Latency
 - i. Enabled = True
 - ii. Low Latency Buffer Option = Auto Buffer Management
 - b. SPD
 - i. Enabled = True
 - ii. Delay = 3000ms
 - iii. Sync = 500ms
 - iv. Too much sync = 10000ms
3. Navigate back to the main screen
4. Tap **Go to URL**
5. In the **URL** field, enter the playback URL in the last step of the [EdgeCaster Configuration](#)
6. Tap **PLAY** at the to start the stream



To configure NexPlayer on **HTML5**:

1. Copy the following code into a file named NexPlayer.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0,
maximum-scale=1.0, user-scalable=no" />

  <title>NexPlayer</title>
  <style type="text/css">
    #player_container {
      width: 90%;
      margin: auto;
      padding-top: 50.625%; /* 16:9 Aspect Ratio 56.25 * 0.9 */
      position: relative;
    }
    @media (min-width: 75rem) {
      #player_container {
        width: 50%;
        padding-top: 28.125%; /* 16:9 Aspect Ratio 56.25 * 0.5 */
      }
    }
    h1 {
      text-align: center;
    }
    #player {
      background-color: black;
      position: absolute;
      top: 0px;
      width: 100%;
      height: 100%;
    }
    #warning {
      background-color: red;
      text-align: center;
      display: none;
    }
  </style>
</head>
```

```

<body>
  <h1>NexPlayer HTML5</h1>
  <div id="warning">
    <h1>Unsupported protocol</h1>
    <h3>Loading HTML using the file protocol can't be supported. Please use a <a
href="https://NexPlayer.github.io/getting_started.html#explanation">server</a>
(HTTP/HTTPS protocol).</h3>
  </div>
  <div id="player_container">
    <div id="player"></div>
  </div>

  <script src="https://NexPlayer.NexPlayersdk.com/latest/NexPlayer.js"></script>
  <script type="text/javascript">
    NexPlayer.Setup({
      key: "[LICENSE KEY]",
      div: document.getElementById('player'),
      lowLatency: true, // toggle on/off low latency apis
      lowLatencyLiveDelay: 3.0, // The desired latency to maintain
      src: '[HLS/DASH VIEWING URL]',

    });

  </script>
</body>
</html>

```

2. After **key**, replace **[LICENSE KEY]** with the HTML5 license associated with your NexPlayer account
3. After **src**, replace **[HLS/DASH VIEWING URL]** with the URL in the last step of the [EdgeCaster Configuration](#)
4. Save the file
5. Load the NexPlayer.html file using a web browser (if not the default when double-clicked, it is usually an option when right-clicked)

Test Results and Observations

All test results below were observed as equivalent for both Single Bitrate and Multiple Bitrate tests.

iOS and HTML5 do not yet support automatic switching of bitrates mid-stream (ABR), so manual switching was used in those tests.

Quantitative test results

Test	Description	DASH / HLS	Platform	Measurements / Comments
ULL	Measures how low latency can be configured throughout the workflow.	DASH	Android	2.3 seconds
			iOS	2.8 seconds
			HTML5	2.7 seconds
		HLS	Android	2.5 seconds
			iOS	3 seconds
			HTML5	>5 seconds
Sync	Measures how closely two devices can be synchronized when playing back on DASH/HLS	DASH	Android	Within 1 second
			iOS	Within 1 second
			HTML5	N/A - See Qualitative results
		HLS	Android	Within 1 second
			iOS	Within 1 second
			HTML5	N/A - See Qualitative results

Qualitative test results

#	Test	Description	DASH / HLS	Pass / Fail		Measurements / Comments
1	Sync	Measures how closely two devices can be synchronized when playing back on DASH/HLS	DASH	Android	Pass	
				iOS	Pass	
				HTML5	N/A	No option for Sync settings
		HLS	Synchronization within 1 second is considered a pass	Android	Pass	
				iOS	Pass	
				HTML5	N/A	No option for Sync settings
2	TS	Verifies if TS playback is supported.	DASH	Android	N/A	Expected result, support not required (not standard)
				iOS	N/A	
				HTML5	N/A	
		DASH	Successful playback of TS is considered a pass	Android	Pass	
				iOS	Pass	
				HTML5	Pass	
3	CMAF	Verifies if CMAF playback is supported	DASH	Android	Pass	
				iOS	Pass	
				HTML5	Pass	
		HLS	Successful playback of CMAF is considered a pass	Android	Pass	
				iOS	Pass	
				HTML5	Pass	
4	ULL	Measures how low latency can be configured throughout the workflow. Stable latency of lower than 3 seconds is considered a pass	DASH	Android	Pass	
				iOS	Pass	
				HTML5	Fail	Observed latency of 2.7 seconds, but failure due to unstable playback. Playback would pause/buffer after a few seconds and does not recover Unable to observe Low Latency even when LiveDelay value increased to 10 seconds
			HLS	Android	Pass	
				iOS	Pass	
				HTML5	Fail	ULL did not appear to work when Low Latency enabled
5	H.264	Verifies if H.264 is supported on playback	DASH	Android	Pass	
				iOS	Pass	

		Successful playback of H.264 is considered a pass		HTML5	Pass	
			HLS	Android	Pass	
				iOS	Pass	
				HTML5	Pass	
6	H.265	Verifies if H.265 is supported on playback	DASH	Android	Pass	
				iOS	Pass	
				HTML5	N/A	N/A since H.265 is dependent on the hardware supporting it. This is likely not anything NexPlayer can easily resolve due to it being more of a hardware issue than a player client software issue.
		Successful playback of H.265 is considered a pass	HLS	Android	Pass	
				iOS	Pass	
				HTML5	N/A	N/A since H.265 is dependent on the hardware supporting it. This is likely not anything NexPlayer can easily resolve due to it being more of a hardware issue than a player client software issue.
13	Chrome	Successful non-ULL playback is considered a pass		Windows	Pass	
				MacOS	Pass	
				Linux	Pass	
14	Safari	Successful non-ULL playback is considered a pass		Windows	N/A	
				MacOS	Pass	
				Linux	Pass	
15	Firefox	Successful non-ULL playback is considered a pass		Windows	Pass	
				MacOS	Pass	
				Linux	Pass	
16	Opera	Successful non-ULL playback is considered a pass		Windows	Pass	
				MacOS	Pass	
				Linux	Pass	
17	Edge	Successful non-ULL playback is considered a pass		Windows	Pass	
				MacOS	N/A	
				Linux	N/A	

Conclusion/Recommendations

Videon concludes the following:

- NexPlayer produces positive results in Android- and iOS-based workflows that utilize ULL, cross-device synchronization, or both
 - Latency was under 3 seconds
 - Cross-device synchronization was within 1 second
- NexPlayer on HTML5 does not appear to fully support ULL, cross-device synchronization, or both
 - Latency was not under 3 seconds
 - Cross-device synchronization was not within 1 second
- In general, video quality appears acceptable (i.e. no issues identified on the player end), even when in ULL conditions (less than 3 seconds).
- Regarding the recommendation of this workflow, with the current state of test results, Videon can only recommend this workflow for the following (for both Single Bitrate and Multiple Bitrate workflows):
 - Non-ULL HLS/DASH on all platforms
 - ULL HLS/DASH workflows on Android and iOS, which can include cross-device synchronization
 - ULL DASH workflows on HTML which do not include cross-device synchronization

Videon recommends the following actions to improve performance or support new features:

- Videon and NexPlayer to investigate the reasons for failure on HTML5-based workflows. Action items for each/either party should be generated as a result of the below investigations.
 - Optimize HTML5 HLS ULL to under 3 seconds of latency
 - Optimize HTML5 synchronization to within one second across devices
 - Ensure HTML5 ULL and synchronization work simultaneously