The Wowza Streaming Cloud™ service allows you to deliver live streams to viewers over multiple protocols: RTMP, RTSP, WOWZ, and HLS directly from the transcoder, and HLS from a CDN endpoint. If latency is a concern, direct RTMP, RTSP, or WOWZ playback is the faster option, but the number of concurrent viewers who can watch your stream this way is limited. HLS delivery from a CDN endpoint, on the other hand, allows you to reach more viewers—and more geographically dispersed viewers—but the latency is greater.

This article explains how to use the Wowza Streaming Cloud REST API to deliver streams with reduced latency and a more scalable deployment. The delivery methods used in this article provide an example. You can choose to just use one, or customize the destinations and settings in whatever way makes sense for your streaming needs.

**Note:** For information about ultra low latency streaming, see the [Get started with ultra low latency streaming using the Wowza Streaming Cloud REST API](#).

### About latency over direct playback URLs and HLS

A lower-latency workflow can be used to deliver a stream directly from the Wowza Streaming Cloud transcoder (the origin server) to your audience over RTMP, RTSP, or WOWZ direct playback URLs. Under optimal conditions—the viewer is close to your broadcast location, network conditions are good, and your source encoding settings are optimal—delivering a stream directly from the Wowza Streaming Cloud transcoder to viewers offers latency as low as 1 second. The advantage of this reduced latency, however, may be offset by the facts that the transcoder is limited to 100 maximum connections and latency increases the farther the viewer is located from the transcoder.
A second option for lower-latency streaming is to use a Wowza CDN on Fastly stream target or Wowza CDN on Akamai - HLS stream target that parses the stream into small media segments, or short chunks. By reducing the HLS media segment size from the default 10 seconds to 2 seconds, you can achieve a latency of as low as 8 to 10 seconds. Both types of Wowza CDN targets have no restrictions on the number of viewers who can access the stream, and they are widely distributed, with servers that are close to viewers wherever they watch. And because HLS is based on HTTP, it performs well on all kinds of networks.

**Note:** Although delivering shorter audio and video file segments can reduce latency, shorter file segments also create additional network overhead for the client. If the client doesn’t have enough bandwidth, playback may stall, which results in a poor viewing experience.

Create a reduced-latency transcoder and a passthrough output for direct RTMP playback

Start by using the Wowza Streaming Cloud REST API to create a transcoder that receives video from a source encoder and delivers a passthrough stream to your targets. Use the following transcoder parameters for your reduced-latency workflow.

Reduced-latency transcoder parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>broadcast_location</td>
<td>string</td>
<td>Specify the region that’s closest to where your stream originates. Valid values are asia_pacific_australia, asia_pacific_india, asia_pacific_japan, asia_pacific_taiwan, eu_belgium, eu_germany, eu_ireland, south_america_brazil, us_central_iowa, us_east_s_carolina,</td>
</tr>
</tbody>
</table>
us_east_virginia, us_west_california, and us_west_oregon.

**Notes:**
- *asia_pacific_taiwan, eu_belgium,*
  *us_central_iowa,* and
  *us_east_s_carolina* don’t allocate dedicated GPU resources to 4K, 24x7 streams. As a result, running 4K streams in the 24x7 billing mode at these locations is not recommended.
- Region availability depends on your Wowza Streaming Cloud plan provider.
  *asia_pacific_s_korea* and *eu_ireland* broadcast locations are available to direct Wowza subscribers only.

<table>
<thead>
<tr>
<th>buffer_size</th>
<th>integer</th>
<th>Specify <strong>0</strong>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>low_latency</td>
<td>Boolean</td>
<td>Specify <strong>true</strong> to speed the time it takes to decode and deliver video data to the player by turning off the sort packet buffer.</td>
</tr>
<tr>
<td>play_maximum_connections</td>
<td>integer</td>
<td>The number of connections that can access the transcoder’s RTMP playback URL. The default is <strong>10</strong>. Specify a value up to <strong>100</strong>.</td>
</tr>
<tr>
<td><strong>stream_smoother</strong></td>
<td><strong>Boolean</strong></td>
<td>Use the default, <strong>false</strong>. (Stream smoothing is not available when the buffer is 0.)</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>transcoder_type</strong></td>
<td><strong>string</strong></td>
<td>Specify <strong>passthrough</strong>.</td>
</tr>
</tbody>
</table>

**Note:** Passthrough streams require a subscription that supports them, such as the Wowza Streaming Cloud API pricing plan.

**Note:** For information on other transcoder parameters, see the Wowza Streaming Cloud REST API Reference Documentation.

Example request and response

Create the transcoder:

```bash
curl -X POST \
-H 'Content-Type: application/json' \
-H "wsc-api-key: ${WSC_API_KEY}" \
-H "wsc-access-key: ${WSC_ACCESS_KEY}" \
-d '{
  "transcoder": {
    "billing_mode": "pay_as_you_go",
    "broadcast_location": "eu_belgium",
    "buffer_size": 0,
    "delivery_method": "push",
    "low_latency": true,
    "name": "MyLowLatencyTranscoder",
    "play_maximum_connections": 100,
    "protocol": "rtmp",
    "stream_smoother": false,
    "transcoder_type": "passthrough"
  }
}' "${WSC_HOST}/api/${WSC_VERSION}/transcoders/
```

The command creates a passthrough transcoder with an id parameter **1234abcd** in
Create the passthrough output:

Note: For more information on passthrough output parameters, see Create an output for the transcoder.
Add a stream target

Next, add a Wowza CDN on Fastly stream target that delivers the stream to geographically distributed Wowza CDN endpoints. You can also use a Wowza CDN on Akamai stream target for this step.

Wowza CDN on Fastly stream target parameter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data</th>
<th>Description</th>
</tr>
</thead>
</table>

The command creates the output with an id parameter (5678efgh in this example) but no targets ("targets": []). The details of the configured output are listed in the response, which should look something like this:
<table>
<thead>
<tr>
<th>Type</th>
<th>name</th>
<th>string</th>
</tr>
</thead>
</table>

A descriptive name for the stream target such as **MyLowLatencyHLStarget**.

Example request and response

Create the target:

```bash
curl -X POST \
-H 'Content-Type: application/json' \
-H "wsc-api-key: ${WSC_API_KEY}" \
-H "wsc-access-key: ${WSC_ACCESS_KEY}" \
-d '{
    "stream_target_fastly": {
        "name": "MyLowLatencyHLStarget"
    }
}' '${WSC_HOST}/api/${WSC_VERSION}/stream_targets/fastly'
```

The command creates the target with an `id` parameter (**3456mnop** in this example). The details of the configured target are listed in the response, which should look something like this:

```json
{
    "stream_target_fastly": {
        "id": "3456mnop",
        "name": "MyLowLatencyHLStarget",
        "state": "activated",
        "stream_name": "6f86005b",
        "playback_url": "https://[subdomain].wowza.com/1/abc2TnJwZEpwXYz/6f86005b/hls/live/playlist.m3u8",
        "token_auth_enabled": false,
        "token_auth_playlist_only": false,
        "geoblock_enabled": false,
        "geoblock_by_location": "disabled",
        "geoblock_ip_override": "disabled",
        "force_ssl_playback": false,
        "created_at": "2020-01-28T11:01:45.044",
        "updated_at": "2020-01-28T11:01:45.044"
    }
}
```

Reduce the chunk size of the stream target
Reduce the duration of the stream’s HLS segments by specifying a small chunk size property for the target:

```
curl -X POST \\n-H 'Content-Type: application/json' \\n-H "wsc-api-key: ${WSC_API_KEY}" \\n-H "wsc-access-key: ${WSC_ACCESS_KEY}" \\n-d '{  "property": {  "key": "chunkSize",  "section": "hls",  "value": "2"  }}' "${WSC_HOST}/api/${WSC_VERSION}/stream_targets/[stream_target_id]/properties"
```

The command assigns a chunk size (media segment size) of 2 to the target. The details of the target’s property are listed in the response, which should look something like this:

```json
{
    "property": {
        "key": "chunkSize",
        "section": "hls",
        "value": "2"
    }
}
```

**Assign the stream target to the output**

Assign the Wowza CDN on Fastly stream target to the output:

```
curl -X POST \\n-H 'Content-Type: application/json' \\n-H "wsc-api-key: ${WSC_API_KEY}" \\n-H "wsc-access-key: ${WSC_ACCESS_KEY}" \\n-d '{  "output_stream_targets": {  "stream_target_id": "3456mnop",
        "use_stream_target_backup_url": false
    }}' "${WSC_HOST}/api/${WSC_VERSION}/transcoders/[transcoder_id]/outputs/[output_id]/output_stream_targets"
```
Get the transcoder's direct playback URLs

Get the transcoder's direct playback URLs so that viewers can play from the transcoder via RTMP, RTSP, WOWZ, or even HLS:

```bash
curl -X GET 
-H "wsc-api-key: ${WSC_API_KEY}" 
-H "wsc-access-key: ${WSC_ACCESS_KEY}" 
"${WSC_HOST}/api/${WSC_VERSION}/transcoders/[transcoder_id]"
```

More resources

- Play streams directly from a transcoder with the Wowza Streaming Cloud REST API
- Best practices for setting the segment duration
- Encoding best practices for Wowza Streaming Cloud