Learn how to use the Wowza Streaming Cloud™ REST API to set up a Wowza stream source with a live stream or transcoder. This allows Wowza Streaming Cloud to automatically detect broadcast location, automatically start a stream when the video source starts, and automatically stop a stream after the video source disconnects.

Associate a Wowza stream source with a live stream

If you prefer the live stream workflow, which automatically creates output renditions and a stream target for you, create a live stream that receives your encoded video at a stream source.

1. Create a live stream.

Live stream properties

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aspect_ratio_height</td>
<td>integer</td>
<td>The height, in pixels, of the output rendition. Should correspond to the aspect ratio (widescreen or standard) of the video source and be divisible by 8. Use the default, 1080.</td>
</tr>
<tr>
<td>aspect_ratio_width</td>
<td>integer</td>
<td>The width, in pixels, of the output rendition. Should correspond to the aspect ratio (widescreen or standard) of the video source and be divisible by 8. Use the default, 1920.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Data Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>billing_mode</td>
<td>string</td>
<td>The billing mode for the stream. Use the default, pay_as_you_go.</td>
</tr>
<tr>
<td>broadcast_location</td>
<td>string</td>
<td>Enter any valid broadcast location value. Wowza Streaming Cloud will override this value to use the closest ingest point based on a DNS query. For a list of valid values, see the <a href="#">REST API reference documentation</a>.</td>
</tr>
<tr>
<td>delivery_method</td>
<td>string</td>
<td>The method you’re using to deliver the video stream to the transcoder. Use <strong>cdn</strong>.</td>
</tr>
<tr>
<td>encoder</td>
<td>string</td>
<td>The video source for your live stream. Use <strong>other_rtmp</strong>.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the live stream. Enter an alphanumeric string (maximum 200 characters).</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
<td>The transport protocol you’re using to send the encoded stream to the transcoder. Use <strong>rtmp</strong>.</td>
</tr>
<tr>
<td>transcoder_type</td>
<td>string</td>
<td>Specify the default, <strong>transcoded</strong>. You can alternatively use passthrough, depending on your needs and the functionality available at your broadcast location.</td>
</tr>
<tr>
<td>use_stream_source</td>
<td>boolean</td>
<td>If <strong>true</strong>, uses a stream source to deliver the stream to Wowza Streaming Cloud. The default, <strong>false</strong>, pushes directly to Wowza Streaming Cloud. Use <strong>true</strong> to receive the encoded video at a stream source.</td>
</tr>
</tbody>
</table>

**Notes:**
- Optionally, you can set `idle_timeout` to determine a specific amount of idle time, in seconds, before the transcoder automatically shuts
Example request and response

Create a live stream:

curl -X POST \
-H "Content-Type: application/json" \
-H "wsc-api-key: ${WSC_API_KEY}" \
-H "wsc-access-key: ${WSC_ACCESS_KEY}" \
-d '{
    "live_stream": {
        "aspect_ratio_height": 1080,
        "aspect_ratio_width": 1920,
        "billing_mode": "pay_as_you_go",
        "broadcast_location": "us_west_oregon",
        "encoder": "other_rtmp",
        "delivery_method": "cdn",
        "name": "Live stream for Wowza Stream Source",
        "transcoder_type": "transcoded",
        "use_stream_source": "true"
    }
} '${WSC_HOST}/api/${WSC_VERSION}/live_streams'

This creates a live stream with outputs and a stream target and automatically associates it with a Wowza stream source. The response should look something like this:
Associate a Wowza stream source with a transcoder

If you prefer the transcoder workflow, which is more modular, you can create a transcoder at the same time as you associate it with a Wowza stream source. First

```json
{
   "live_stream": {
      "id": "abcwksvr",
      "name": "Live Stream for Wowza Stream Source",
      "transcoder_type": "transcoded",
      "billing_mode": "pay_as_you_go",
      "broadcast_location": "us_west_california",
      "recording": false,
      "closed_caption_type": "none",
      "low_latency": false,
      "encoder": "other_rtmp",
      "delivery_method": "cdn",
      "target_delivery_protocol": "hls-https",
      "use_stream_source": true,
      "aspect_ratio_width": 1920,
      "aspect_ratio_height": 1080,
      "delivery_protocols": [
         "rtmp",
         "rtsp",
         "wowz",
         "hls"
      ],
      "source_connection_information": {
         "primary_server": "rtmp://6d3e06.dev.entrypoint.cloud.wowza.com/app-a61e",
         "host_port": 1935,
         "stream_name": "241b5fcf",
         "disable_authentication": false,
         "username": "client2",
         "password": "8ea686ba"
      },
      ...
      "stream_targets": [
         {
            "id": "htwtxqxy"
         }
      ],
      "direct_playback_urls": {...},
      "created_at": "2019-01-22T20:14:41.000Z",
      "updated_at": "2019-01-22T20:14:42.000Z"
   }
}
```
add the stream source and then configure a transcoder to use it.

**Add a Wowza stream source**

1. Create a Wowza stream source with the Wowza Streaming Cloud REST API.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the Wowza stream source. Enter an alphanumeric string that is short (maximum 200 characters) and descriptive.</td>
</tr>
</tbody>
</table>

**Example request and response**

Create a Wowza stream source:

```bash
curl -X POST \
-H "Content-Type: application/json" \
-H "wsc-api-key: ${WSC_API_KEY}" \
-H "wsc-access-key: ${WSC_ACCESS_KEY}" \
-d '{
  "stream_source_wowza": {
    "name": "Auto-starting stream source"
  }
}' "${WSC_HOST}/api/${WSC_VERSION}/stream_sources/wowza"
```

This creates a Wowza stream source with source connection details. Note the stream source ID. The response should look something like this:

```json
{
  "stream_source_wowza": {
    "id": "def0vjgyv",
    "name": "Auto-starting stream source",
    "primary_url": "rtmp://origin.cdn.wowza.com:1935/live",
    "stream_name": "012p2abcKS3Vq93MGJIF6V5tIYS9560a",
    "state": "stopped",
    "created_at": "2018-12-12T15:37:55.000Z",
    "updated_at": "2018-12-12T15:37:55.000Z"
  }
}
```
**Note:** Wowza stream sources don’t allow source authentication.

Add a transcoder

1. Create a transcoder with a `delivery_method` of **cdn**, and associate it with the stream source ID.

### Transcoder properties

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>billing_mode</code></td>
<td>string</td>
<td>The billing mode for the stream. Use the default, <strong>pay_as_you_go</strong>.</td>
</tr>
<tr>
<td><code>broadcast_location</code></td>
<td>string</td>
<td>Enter any valid broadcast location value. Wowza Streaming Cloud will override this value to use the closest ingest point based on a DNS query. For a list of valid values, see the REST API reference documentation.</td>
</tr>
<tr>
<td><code>delivery_method</code></td>
<td>string</td>
<td>The method you're using to deliver the video stream to the transcoder. Use <strong>cdn</strong> so the video stream will push to a stream source.</td>
</tr>
<tr>
<td><code>idle_timeout</code></td>
<td>integer</td>
<td>Set an <code>idle_timeout</code> value that determines the amount of idle time, in seconds, before the transcoder automatically shuts down. Idle time begins after the video source is disconnected at the end of the live broadcast. The default is <strong>1200</strong> (20 minutes).</td>
</tr>
<tr>
<td><code>name</code></td>
<td>string</td>
<td>The name of the transcoder. Enter an alphanumeric string (maximum 200 characters).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The transport protocol you’re using to...</td>
</tr>
<tr>
<td>Parameter</td>
<td>Data Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
<td>send the encoded stream to the transcoder. Use rtmp.</td>
</tr>
<tr>
<td>stream_source_id</td>
<td>string</td>
<td>The alphanumeric string that identifies the stream source that you want to use to deliver the stream to the transcoder. Use the ID of the Wowza stream source you just created.</td>
</tr>
<tr>
<td>transcoder_type</td>
<td>string</td>
<td>Specify the default, transcoded. You can alternatively use passthrough, depending on your needs and the functionality available at your broadcast location.</td>
</tr>
</tbody>
</table>

## Example request and response

**Create a 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": "us_west_oregon",
        "delivery_method": "cdn",
        "name": "Transcoder for Wowza Stream Source",
        "protocol": "rtmp",
        "stream_source_id": "def0vjqv",
        "transcoder_type": "transcoded"
    }
}' ${WSC_HOST}/api/${WSC_VERSION}/transcoders
```

This creates a transcoder without outputs or stream targets. The transcoder is associated with the existing Wowza stream source, and Wowza Streaming Cloud will override the broadcast_location value. The response should look something like this:
Complete the transcoder by adding outputs and stream targets. For instructions, see one of the following articles, depending on whether you’re creating an adaptive bitrate stream or passthrough stream:

- **Create an ABR stream and send it to a target with the Wowza Streaming Cloud REST API**
- **Pass a stream through the transcoder to a target with the Wowza Streaming Cloud REST API**

### Test your connection

1. Set up your video source using the `primary_url` and `stream_name` from the Wowza stream source details. The primary URL is the ingest location. For example:

   ```
   rtmp://origin.cdn.wowza.com:1935/live
   ```

   The `stream_name` is the alphanumeric stream name generated by Wowza Streaming Cloud to identify the specific stream. For example:
Refer to documentation for your specific encoder to note where to input the stream settings.

For example, **Address** or **URL** might be:

```
rtmp://origin.cdn.wowza.com:1935/live
```

While **Stream** or **Stream key** might be:

```
012p2abcKS3Vq93MGJIF6V5tIYS9560a
```

2. **Start your video source.**

Wowza Streaming Cloud receives the stream from the source, starts transcoding, creates output renditions for the stream, and sends each output rendition to the stream target assigned to it.

3. **Using the Wowza Streaming Cloud REST API, fetch the state of the transcoder or live stream to confirm that it’s started.**

   Fetch the state of the transcoder:

   ```
curl -X GET \
-H "wsc-api-key: "$WSC_API_KEY" \
-H "wsc-access-key: "$WSC_ACCESS_KEY" \
"https://api.cloud.wowza.com/api/[version]/transcoders/[transcoder_id]/state"
```

   Fetch the state of the live stream:

   ```
curl -X GET \
-H "wsc-api-key: "$WSC_API_KEY" \
-H "wsc-access-key: "$WSC_ACCESS_KEY" \
"https://api.cloud.wowza.com/api/[version]/live_streams/[live_stream_id]/state"
```

4. **(Optional) In the Wowza Streaming Cloud user interface, confirm that the stream is working by looking at the Video Thumbnail on the Overview tab of the transcoder detail page or live stream detail page.**

   **Note:** The API request to fetch a thumbnail URL of a transcoder or live stream does not work for a transcoder with a Wowza stream source.
5. To end your test stream, stop the video source.

Once you disable the video source, the transcoder or live stream continues to run and accrue charges until it reaches the default idle timeout value of 20 minutes. It then stops automatically. You can set the idle timeout value when you create or update the transcoder so that the transcoder stops sooner.

Alternatively, you can use the REST API to stop the transcoder programmatically.

Stop the transcoder:

```
curl -X PUT \\n-H "wsc-api-key: ${WSC_API_KEY}" \\n-H "wsc-access-key: ${WSC_ACCESS_KEY}" \\n"https://api.cloud.wowza.com/api/[version]/transcoders/[transcoder_id]/stop"
```

More resources

- Speed up the connection to a transcoder with the Wowza Streaming Cloud REST API