Web Real-Time Communication (WebRTC) is an open-source project that enables real-time communication of audio, video, and data in web browsers and apps. The Wowza Streaming Cloud™ service supports ingest and transcoding of WebRTC streams through the live stream or transcoder workflow.

**Note:** Source authentication and closed captions aren’t available for WebRTC.

### Create a live stream

The live stream workflow allows you to configure more settings in one API request, while the transcoder workflow allows more modular, custom configuration of settings using multiple API requests. Start with this section if you choose the live stream workflow.

1. Using the Wowza Streaming Cloud REST API, create a live stream.

#### Live stream parameters

<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.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value should correspond to the aspect ratio (widescreen or standard) of the video source and be divisible by 8. Set the aspect ratio of the live stream to match the aspect ratio in your encoder settings.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Data Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>aspect_ratio_width</td>
<td>integer</td>
<td>The width, in pixels, of the output rendition. This value should correspond to the aspect ratio (widescreen or standard) of the video source and be divisible by 8. Set the aspect ratio of the live stream to match the aspect ratio in your encoder settings.</td>
</tr>
<tr>
<td>billing_mode</td>
<td>string</td>
<td>The billing mode for the stream. Specify the default value, pay_as_you_go.</td>
</tr>
<tr>
<td>broadcast_location</td>
<td>string</td>
<td>The region that’s closest to where your stream originates. For a list of valid regions, see the API reference documentation.</td>
</tr>
<tr>
<td>encoder</td>
<td>string</td>
<td>The video source for the live stream. Specify other_webrtc.</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the live stream. Enter an alphanumeric string that is short (maximum 200 characters) and descriptive, for example, MyWebRTCStream.</td>
</tr>
<tr>
<td>transcoder_type</td>
<td>string</td>
<td>The type of transcoder. Specify the default value, transcoded. Wowza Streaming Cloud doesn’t support passthrough streams for WebRTC.</td>
</tr>
<tr>
<td>delivery_method</td>
<td>string</td>
<td>The method you’re using to deliver the source stream to the transcoder. Specify the default value, push. Wowza Streaming Cloud doesn’t support pull or cdn connections for WebRTC.</td>
</tr>
</tbody>
</table>
For a full list of live stream parameters, see Live Streams in the Wowza Streaming Cloud API reference.

Example request and response

curl -X POST \
-H "Content-Type: application/json" \
-H "wsc-api-key: ${WSC_API_KEY}" \
-H "wsc-access-key: ${WSC_ACCESS_KEY}" \
-d '{
    "live_stream": {
        "aspect_ratio_height": 720,
        "aspect_ratio_width": 1280,
        "billing_mode": "pay_as_you_go",
        "broadcast_location": "us_west_california",
        "delivery_method": "push",
        "encoder": "other_webrtc",
        "name": "MyWebRTCStream",
        "transcoder_type": "transcoded"
    }
}' "${WSC_HOST}/api/${WSC_VERSION}/live_streams"

This request creates a live stream with an id parameter, an associated player, and a hosted page. The details of the live stream’s configuration are listed in the response. For WebRTC, you’ll use the source_connection_information to configure the stream.

{  
    "live_stream": {
        "id": "1234abcd",
        "name": "MyWebRTCStream",
        ...
        "encoder": "other_webrtc",
        ...
        "source_connection_information": {
            "sdp_url": "wss://[subdomain].entrypoint.cloud.wowza.com/webrtc-session.json",
            "application": "app-464b",
            "stream_name": "32a5814b"
        },
        ...
    }
}

Related API requests
View the details of a live stream, including the player embed code and hosted page URL:

```bash
curl -X GET \\n-H "wsc-api-key: $(WSC_API_KEY)" \\
-H "wsc-access-key: $(WSC_ACCESS_KEY)" \\
"${WSC_HOST}/api/${WSC_VERSION}/live_streams/[live_stream_id]"
```

Update a live stream's configuration:

```bash
curl -X PATCH \\n-H "Content-Type: application/json" \\
-H "wsc-api-key: $(WSC_API_KEY)" \\
-H "wsc-access-key: $(WSC_ACCESS_KEY)" \\
-d "{
   "live_stream": {
      "name": "MyDifferentLiveStreamName"
   }
}" \\
"${WSC_HOST}/api/${WSC_VERSION}/live_streams/[live_stream_id]"
```

Delete a live stream:

```bash
curl -X DELETE \\n-H "wsc-api-key: $(WSC_API_KEY)" \\
-H "wsc-access-key: $(WSC_ACCESS_KEY)" \\
"${WSC_HOST}/api/${WSC_VERSION}/live_streams/[live_stream_id]"
```

2. After creating the live stream, configure the WebRTC stream. To publish a test stream to Wowza Streaming Cloud, see Test the live stream connection.

Test the live stream connection

The Wowza Streaming Cloud documentation includes a downloadable example that shows how to publish a WebRTC stream to Wowza Streaming Cloud.

**Note:** The WebRTC example must be hosted on a web server (for example, Apache HTTP Server) that uses SSL encryption.

1. Download the WebRTC HTML example from Github.
2. In your file system, navigate to the example files and copy the contents of the html folder to your web server.
3. Open `index.html` in a web browser. Be sure to grant access to your camera and microphone when prompted.

4. Use the `source_connection_information` from the REST API response to configure the stream settings:
   - **Host Server** – Replace `wss://[wowza_subdomain].entrypoint.cloud.wowza.com/webrtc-session.json` with the `sdp_url`.
   - **Application Name** – Replace `webrtc` with the `application_name`.
   - **Stream Name** – Replace `myStream` with the `stream_name`.

5. Start the live stream using the Wowza Streaming Cloud REST API.

   ```
curl -X PUT \
-H "wsc-api-key: ${WSC_API_KEY}" \
-H "wsc-access-key: ${WSC_ACCESS_KEY}" \
"${WSC_HOST}/api/${WSC_VERSION}/live_streams/[live_stream_id]/start"
```

Alternatively, click **Start Live Stream** at the top of the live stream detail page in the Wowza Streaming Cloud user interface.

6. If you’re using the Wowza Streaming Cloud REST API to start the live stream, fetch the state of the live stream to make sure it’s started.

   ```
curl -X GET \
-H "wsc-api-key: ${WSC_API_KEY}" \
-H "wsc-access-key: ${WSC_ACCESS_KEY}" \
"${WSC_HOST}/api/${WSC_VERSION}/live_streams/[live_stream_id]/state"
```

7. On the `index.html` page, click **Publish**.

8. Confirm that the stream is playing.

   a. Fetch a video thumbnail of the stream using the Wowza Streaming Cloud REST API.

   ```
curl -X GET \
-H "wsc-api-key: ${WSC_API_KEY}" \
-H "wsc-access-key: ${WSC_ACCESS_KEY}" \
"${WSC_HOST}/api/${WSC_VERSION}/live_streams/[live_stream_id]/thumbnail_url"
```

   b. View the thumbnail URL in a browser.
c. Alternatively, in the Wowza Streaming Cloud user interface, confirm that the live stream is playing by looking at the **Video Thumbnail** in the **Overview** tab of the live stream detail page.

9. Use the Wowza Streaming Cloud REST API to stop the live stream.

**Stop the live stream:**

```bash
curl -X PUT \
-H "wsc-api-key: ${WSC_API_KEY}" \
-H "wsc-access-key: ${WSC_ACCESS_KEY}" \
"${WSC_HOST}/api/${WSC_VERSION}/live_streams/[live_stream_id]/stop"
```

Alternatively, click **Stop Live Stream** at the top of the live stream detail page in the Wowza Streaming Cloud user interface.

**Create a transcoder**

The transcoder workflow allows more modular, custom configuration of settings using multiple API requests. Start with this section if you choose the transcoder workflow.

1. Using the Wowza Streaming Cloud REST API, create a transcoder.

**Transcoder parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>billing_mode</td>
<td>string</td>
<td>The billing mode for the stream. Specify the default value, <code>pay_as_you_go</code>.</td>
</tr>
<tr>
<td>broadcast_location</td>
<td>string</td>
<td>Specify the region that’s closest to where your stream originates. For a list of valid regions, see the <a href="#">API reference documentation</a>.</td>
</tr>
<tr>
<td>delivery_method</td>
<td>string</td>
<td>The method you’re using to deliver the source stream to the transcoder. Specify <code>push</code>. Wowza Streaming Cloud doesn’t support <code>pull</code> or <code>cdn</code> connections for WebRTC.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Data Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the transcoder. Enter an alphanumeric string that is short (maximum 200 characters) and descriptive, for example, MyWebRTCTranscoder.</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
<td>The transport protocol you’re using to send the encoded stream to the transcoder. Specify webrtc.</td>
</tr>
<tr>
<td>transcoder_type</td>
<td>string</td>
<td>Specify the default value, transcoded. Wowza Streaming Cloud doesn’t support passthrough streams for WebRTC.</td>
</tr>
</tbody>
</table>

**Example request and response**

```
curl -X POST \
-H "Content-Type: application/json" \ 
-H "wsc-api-key: ${WSC_API_KEY}" \ 
-H "wsc-access-key: ${WSC_ACCESS_KEY}" \ 
-d '{
  "transcoder": {
    "billing_mode": "pay_as_you_go",
    "broadcast_location": "us_west_california",
    "delivery_method": "push",
    "name": "MyWebRTCTranscoder",
    "protocol": "webrtc",
    "transcoder_type": "transcoded"
  }
}' "${WSC_HOST}/api/${WSC_VERSION}/transcoders"
```

This request creates a transcoder with an id parameter, but no outputs or stream targets. The details of the transcoder’s configuration are listed in the response.
2. Complete the transcoder by adding output renditions and stream targets. For more information, see [Create an ABR stream and send it to a target with the Wowza Streaming Cloud REST API](#).

**Note:** Passthrough video and passthrough audio output renditions aren’t available for WebRTC.

---

**Test the transcoder connection**

The Wowza Streaming Cloud documentation includes a downloadable example that shows how to publish a WebRTC stream to Wowza Streaming Cloud.

**Notes:**
- The WebRTC example must be hosted on a web server (for example, [Apache HTTP Server](http://httpd.apache.org)) that uses SSL encryption.
- In order for the example to work properly, the transcoder `stream_extension` must be set to `.json`.

1. Download the [WebRTC HTML example](https://github.com/wowza/wowza-cloud-streaming-examples/tree/master/WEBRTC) from Github.
2. In your file system, navigate to the example files and copy the contents of the `html` folder to your web server.
3. Open `index.html` in a web browser. Be sure to grant access to your camera and microphone when prompted.

4. Use the transcoder information from the REST API response to configure the stream settings:

   - **Host Server** – Replace `[wowza_subdomain].entrypoint.cloud.wowza.com` with the `domain_name`.
   - **Application Name** – Replace `webrtc` with the `application_name`.
   - **Stream Name** – Replace `myStream` with the `stream_name`.

5. Start the transcoder using the Wowza Streaming Cloud REST API.

   ```bash
   curl -X PUT \
   -H "wsc-api-key: $WSC_API_KEY" \
   -H "wsc-access-key: $WSC_ACCESS_KEY" \
   "$WSC_HOST/api/$WSC_VERSION/transcoders/[transcoder_id]/start"
   ```

   Alternatively, click **Start Transcoder** at the top of the transcoder detail page in the Wowza Streaming Cloud user interface.

6. If you’re using the Wowza Streaming Cloud REST API to start the transcoder, fetch the state of the transcoder to make sure it’s started.

   ```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]/state"
   ```

7. On the `index.html` page, click **Publish**.

8. Confirm that the stream is playing.

   a. Fetch a video thumbnail of the stream using the Wowza Streaming Cloud REST API.

      ```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]/thumbnail_url"
      ```

   b. View the thumbnail URL in a browser.
c. Alternatively, in the Wowza Streaming Cloud user interface, confirm that the transcoder is playing by looking at the Video Thumbnail in the Overview tab of the transcoder detail page.

9. Use the Wowza Streaming Cloud REST API to stop the transcoder.

Stop the transcoder:

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

Alternatively, click Stop Transcoder at the top of the transcoder detail page in the Wowza Streaming Cloud user interface.

More resources

- Wowza Streaming Cloud REST API reference documentation
- Connect a WebRTC stream to Wowza Streaming Cloud