Allissa Hertz
Writing Sample

This writing sample is a how-to document I wrote for Wowza Streaming Cloud. This represents original writing. I was the only editor for this article. I gathered the information for this article by tracking the development of the functionality and using details from user stories. I wrote the cURL code samples. The JavaScript code samples were provided by engineers, and I tested them along with QA, using a small NodeJS website I created. I also changed the variables to be more descriptive and obscure private information from the original code the developer sent over.

This article was part of a series of articles for the release of Real-Time Streaming at Scale in the Fall of 2021. I oversaw the development of Real-Time Streaming at Scale from brainstorm to delivery. This included adding new API endpoints, writing UI articles, a conceptual article that explained the difference between WebRTC and Real-Time Streaming at Scale, and editing and restyling a Java SDK reference website.

Deliver real-time streams with the Wowza Streaming Cloud REST API

Wowza Streaming Cloud™ Real-Time Streaming at Scale provides half-second latency to all your viewers, no matter where they are. Real-time streaming is perfect for interactive use cases like video chats, auctions, e-sports, fitness, e-commerce, gambling, and more.

If your audience is fewer than 300 viewers or you want to stream in near real-time alongside other delivery protocols, you may choose to use our standard WebRTC solution.

Note: Real-Time Streaming at Scale is available only through the 1.7 version of the Wowza Streaming Cloud REST API or later.

Streaming using Real-Time Streaming at Scale requires that you provision the stream. The stream can be supplied from Wowza Streaming Cloud using the UI or API. See Deliver real-time streams to viewers with Wowza Streaming Cloud for instructions on how to use the Wowza Streaming Cloud UI to create a Real-Time stream.

We have three options for configuring your streaming publish and playback clients: the Javascript SDK for Real-Time Streaming, OBS for Real-Time Streaming at Scale, and RTMP. You need to create and host a viewer page using HMTL and Javascript to use Real-Time Streaming at Scale with any configuration.

Before you start

You should be familiar with the following concepts:

- **API authentication methods**. Use HMAC authentication for production environments. For testing or proof of concept purposes only, use API key and access key authentication.
- **Environment variables**. We use environment variables for the API version and your API and access keys in the cURL API request examples in this topic to make it easier for you to copy, paste, and run commands in your Terminal or Command Prompt window. If you don't set environment variables for these values, you'll need to manually enter the correct values in the code samples throughout this tutorial. See Tools for testing the API for instructions.

- **Real-time streaming workflows**. See About real-time streaming for more information.
- **HTML and Javascript**.

You should have access to the following items:

- A **RTS@S license**. Contact sales@wowza.com for more information. To enable and purchase capacity for Real-Time Streaming at Scale for your account and access the /real_time operations, contact 720.279.8163 or schedule a call.

1. Create a real-time stream

Create a real-time stream by sending a `POST` request to the `/real_time` endpoint.

You can use the following sample request, making sure to:

- Set `name` to a descriptive name for your real-time stream.
- Set `enable_secure_viewer` to true and set an `expires_on` date if you would like to add a security token that must be passed by viewers for playback.

  Note: `enable_secure_viewer` is available only through the 1.8 (beta) version of the Wowza Streaming Cloud REST API or later.

- Change any values unique to your broadcast, using the API reference documentation as a resource.
- See the Javascript SDK reference for additional configuration options.

Sample request

```
curl -X POST \
-H "Content-Type: application/json" \
-H "wsc-api-key: ${WSC_API_KEY}" \
-H "wsc-access-key: ${WSC_ACCESS_KEY}" \
-d '{
   "real_time_stream": {
     "name": "MyRealTimeStream",
     "enable_secure_viewer": false
```

```
    }
}' "${WSC_HOST}/api/${WSC_VERSION}/real_time"
```

Sample response

The response includes:

- An `id` and a `stream_name` that you can use to configure the viewer page.


```
{
  "real_time_stream": {
    "id": "2adffc17",
    "name": "MyRealTimeStream",
    "stream_name": "8d304b93f1684320a54f2798666eeca7",
    "token": "97e52731bc21ef66e4c05a8ee1e28b64bf5f9db728573d94e690277cea9215bc",
    "subscribe_token": "50e161bfa42fbd581a0dfe5f632596b86c2c577a56bd439b38f8c904aabad
04d",
    "rtmp_url": "rtmp://[primary_server]:[host_port]/[sub-domain]/[stream_name]?[toke
n]",
    "enable_secure_viewer": true,
    "state": "active",
    "created_at": "2021-06-30T18:02:20.00Z",
    "updated_at": "2021-06-30T20:03:16.00Z"
  }
}
```


Wowza Streaming Cloud creates a real-time stream and provides a **token** property for publishing security, a **subscribe_token** property for viewing security, and **stream_name** you will need to configure your source.

2. Configure your video source

Now that you have created your stream, you'll need to configure your real-time stream using the Javascript SDK.

The Javascript SDK lets you build your own custom publication page. We provide sample code to get you started.

You can also use NPM to build your publication and viewer pages. See the Javascript SDK reference for more information.

Use the publisher page sample code, making sure to:

- Set 'my-stream-name' to the `stream_name` generated by Wowza for your real-time stream, for example: *8d304b93f1684320a54f2798666eeca7*
- Set 'my-publish-token' to the `token` generated by Wowza for your real-time stream, for example: *97e52731bc21ef66e4c05a8ee1e28b64bf5f9db728573d94e690277cea9215bc*
- See the [Javascript SDK reference] for additional configuration options.

```
<html>
  <head>
    <script src='https://www.wowza.com/downloads/rts-sdk/wowza.umd.js'></script>
  </head>
  <body>
    <div>
        <h1>Publish</h1>
        <video autoplay playsinline controls muted id="my-video" width="1280" height=
"720"></video>
        <script>
            const wowza = window.wowza
            const streamID = 'my-stream-name'
            const tokenID = 'my-publish-token'

            //Define callback for generate new tokens
            const tokenGenerator = () => wowza.Director.getPublisher({
                token: tokenID,
                streamName: streamID
              })

            //Create a new instance
            const wowzaPublish = new wowza.Publish(streamID, tokenGenerator)

            //Get User camera and microphone
            navigator.mediaDevices.getUserMedia({ audio: true, video: true }).then(
                (mediaStream) => {
                    //Publishing Options
                    const broadcastOptions = {
                        mediaStream
                    }
```

```
                    document.getElementById('my-video').srcObject = mediaStream

                    //Start broadcast
                    try {
                        wowzaPublish.connect(broadcastOptions)
                    } catch (e) {
                        console.log('Connection failed, handle error', e)
                    }
                }
            )
        </scrip>
    </div>
  </body>
</html>
```

3. Configure your viewer page

The Javascript SDK lets you build your own custom viewer page. We provide sample code to get you started.

Use this viewer page sample code, making sure to:

- Set 'my-stream-name' to the *stream_name* for the stream, for example: *11ad1f62a39a4287895d13ac28c0a820*

```
<html>
  <head>
        <script src='https://www.wowza.com/downloads/rts-sdk/wowza.umd.js'></script>
  </head>
  <body>
    <video controls autoplay id="my-video"></video>

    <script>
        const wowza = window.wowza

        // Get Media Element
        const video = document.getElementById('my-video')

        const streamName = "my-stream-name"

        //Define callback for generate new token
```

```
        const tokenGenerator = () => wowza.Director.getSubscriber({
            streamName: streamName
        })

        //Create a new instance
        const wowzaView = new wowza.View(streamName, tokenGenerator, video)

        //Start connection to publisher
        try {
            wowzaView.connect()
        } catch (e) {

            console.log('Connection failed, handle error', e)
        }
    </script>
  </body>
</html>
```

4. Test the connection

1. Start your video source from the publisher page, OBS, or the encoder.
2. Open the file that contains the viewer page code in a web browser to confirm the stream is running.