Learn about the REST API for the Wowza Streaming Cloud™ service, including setting up a request, versioning, authentication, using the sandbox environment, and testing out the API.

**About the REST API**

The Wowza Streaming Cloud REST API (application programming interface) uses the Hypertext Transfer Protocol (HTTP) to request data from Wowza Streaming Cloud servers through requests to API endpoints. You can use the Wowza Streaming Cloud REST API to add live streaming and playback functionality to your applications. The REST API offers complete programmatic control over live streams, transcoders, stream sources, and stream targets. Anything you can do in the Wowza Streaming Cloud UI can also be achieved by making HTTP-based requests to cloud-based servers through the REST API.

**API requests**

Requests in the Wowza Streaming Cloud REST API use JSON syntax for the request body and response. The examples in our articles use a curl command to execute an HTTP method in a Command Prompt or Terminal window. To learn about cURL and other methods for testing out the REST API, see [Tools for testing the API](#).

Curl commands use the following general format:

```
curl -[HTTP method] --[headers] -[parameters] "[resource]"
```

*HTTP method* is the action you’re requesting of the resource. Wowza Streaming Cloud uses these HTTP methods:
### method | Description
--- | ---
POST | POST creates records in the resource’s database. The POST method returns a response that indicates that the request was successful and includes the values of any newly created records.
GET | GET retrieves records from the resource’s database. The GET method returns a response including detailed information about the queried object or objects. You may need values from a GET response in order to use them in a POST or PATCH request.
PATCH | PATCH updates records in the resource’s database. Certain attributes can’t be updated in a PATCH request and must be defined in an initial POST request. The PATCH method returns a response that indicates that the request was successful and includes the values of updated records.
PUT | PUT requests in the Wowza Streaming Cloud REST API start or stop a entity such as a live stream or transcoder. The PUT method response gives information about the entity that was affected.
DELETE | DELETE removes the record from the resource’s database. The delete method returns a response with no content.

*Headers* are information that precede the actual HTTP request. Wowza Streaming Cloud REST API requests require headers in key:value pairs.

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
<th>Where used</th>
</tr>
</thead>
<tbody>
<tr>
<td>wsc-api-key</td>
<td>A unique, 64-digit alphanumeric string. Each Wowza Streaming Cloud account has one API key. The API key can’t be edited or deleted, and it doesn’t expire. See <a href="#">Locate an API key and generate an access key</a> for more information.</td>
<td>For authentication with API keys (recommended for testing and initial development only). In HMAC authentication, <code>wsc-api-key</code> is replaced by a generated signature value, <code>wsc-signature</code>.</td>
</tr>
<tr>
<td></td>
<td>The access key is also a 64-digit</td>
<td></td>
</tr>
<tr>
<td><strong>Parameter</strong></td>
<td>Description</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>wsc-access-key</td>
<td>alphanumeric string, however, each user creates their own in the Wowza Streaming Cloud user interface. You can enable or disable individual access keys, but they don’t expire. See Locate an API key and generate an access key for more information.</td>
<td>For HMAC authentication and authentication with API keys.</td>
</tr>
<tr>
<td>wsc-timestamp</td>
<td>Unix epoch timestamp. See HMAC authentication for more information.</td>
<td>For HMAC authentication.</td>
</tr>
<tr>
<td>wsc-signature</td>
<td>Hex digest-encoded signature string. See HMAC authentication for more information.</td>
<td>For HMAC authentication.</td>
</tr>
<tr>
<td>Content-Type</td>
<td>application/json Specifies that the content sent to the server is in JSON format.</td>
<td>For POST and PATCH HTTP methods only.</td>
</tr>
</tbody>
</table>

**Parameters** refine the request and may correspond to options in the Wowza Streaming Cloud user interface. They can also be known as attributes. In JSON syntax, you define a value for each parameter in key:value pairs.

Example of parameters and their assigned values in a JSON object:

```json
{
  "transcoder": {
    "billing_mode": "pay_as_you_go",
    "broadcast_location": "us_central_iowa",
    "delivery_method": "push",
    "name": "MyPassthruTranscoder",
    "protocol": "rtmp",
    "transcoder_type": "passthrough"
  }
}
```

**Resource**, or base URL, is the location of the server receiving the request. In the Wowza Streaming Cloud REST API production environment, the base URL is

https://api.cloud.wowza.com/api/[version]/
In the Wowza Streaming Cloud REST API sandbox environment, the base URL is

```
https://api.sandbox.cloud.wowza.com/api/[version]/
```

Example of a full API request in curl command syntax:

```
curl -X POST \\
-H "Content-Type: application/json" \\
-H "wsc-api-key: ${WSC_API_KEY}" \\
-H "wsc-access-key: ${WSC_ACCESS_KEY}" \\
-d '{
   "transcoder": {
      "billing_mode": "pay_as_you_go",
      "broadcast_location": "us_central_iowa",
      "delivery_method": "push",
      "name": "MyPassthruTranscoder",
      "protocol": "rtmp",
      "transcoder_type": "passthrough"
   }
}' "https://api.sandbox.cloud.wowza.com/api/${WSC_VERSION}/transcoders"
```

**Note:** The Wowza Streaming Cloud REST API publishing and playback endpoints use dynamic IP addresses. As a result, we can’t provide IP addresses for whitelisting from behind a firewall.

### API versions

The Wowza Streaming Cloud REST API is periodically versioned. Minor updates are iterated using sequential dot numbers; major versions are iterated using sequential whole numbers. Each version is one of these types:

- **Beta** – The beta version contains everything that’s in the current version as well as some features and functionality that are still in development. These work-in-progress features aren’t fully tested and are subject to change. When we’re done testing, we’ll promote the beta version to the current version and then create a new beta with new features. You’re free to use the beta version for testing and evaluation, but beta versions aren’t intended for use in production environments and we caution against using a beta in production. There is only one beta version available at any time.
• **Current** – The current version offers the most complete, up-to-date, tested, and stable code base. We strongly recommend using the current version in your production environment. There is only one current version available at any time.

• **Supported** – A supported version was current at one time but has been replaced by a newer version of the API. Supported versions don’t have the newest features, and may contain features or functions that are outdated and don’t offer the most efficient methods for accomplishing your streaming goals. If you’re using a supported version, we recommend that you upgrade to the current version, as supported versions will ultimately be deprecated.

• **Deprecated** – A deprecated version is out-of-date and can’t be guaranteed to work in production environments. If you’re using a deprecated version, update to the current version at your earliest convenience.

• **Sunset** – A sunset version is no longer accessible. This includes all relevant documentation.

See the [Wowza Streaming Cloud REST API deprecation policy](#) for more information about deprecation and the deprecation schedule.

See [Wowza Streaming Cloud REST API release notes](#) for a detailed changelog of updates to each API version.

---

**API limits**

Reasonable resource limits are in place to prevent excessive usage of streams and stream targets through the Wowza Streaming Cloud REST API. See [Wowza Streaming Cloud REST API limits](#) for details.

For information on limitations of the Wowza Streaming Cloud free trial, see [Trial limitations](#).

---

**Authentication**

For security, all requests to the Wowza Streaming Cloud REST API must include authentication information in the header. There are two methods to authenticate requests:
HMAC authentication

For code deployed to production environments, we recommend hash-based message authentication code (HMAC) for increased security. In this form of authentication, the API key is a private, secret key. It is known to you and the Wowza Streaming Cloud service but never sent directly in an API request.

To perform HMAC authentication, you use a timestamp, the request path, and your secret API key to generate a signature, which you then encrypt using a secure hashing algorithm (SHA-256) with digest encryption and hex encoding. You pass the encrypted signature into the request as a header value along with the timestamp and your access key. When the server receives the request, Wowza Streaming Cloud uses the header values you sent and the secret API key value to regenerate the signature with the secure hashing algorithm, and, if the values match, performs the request.

For HMAC authentication, you'll need:

- An API key
- An access key
- The request path for your API request, without the protocol or query parameters. An initial forward slash (/) is required, and a final forward slash is not allowed. For example, if your initial path is

  https://api.cloud.wowza.com/api/v1.4/live_streams?
  page=1&per_page=10

  it becomes

  /api/v1.4/live_streams

- A timestamp matching these conditions:
  - A Unix epoch value representing the current time
  - A value within 15 seconds of the server clock, or the signature is considered invalid
  - The same timestamp value that’s used to generate the signature
First, you’ll locate your API key and generate an access key. Then you’ll generate and encrypt a signature for the API request.

**Locate an API key and generate an access key**

**Generate and encrypt a signature for the request**

Using the following code examples as a guide, generate the signature, encrypt it, and pass it into a `wsc-signature` header value.

In the code examples, we first generate a signature data string consisting of the timestamp, request path, and api key, delimited by a colon (`:`).

For example:

If `timestamp = 1542064908` and `request_path = /api/v1.4/live_streams` and `api_key = mysupersecretkey`
then `data = 1542064908:/api/v1.4/live_streams:mysupersecretkey`

We then use the signature data and the API key to create a hex string that is HMAC-encoded with SHA256 digest.

Ruby example:
require 'openssl'

# Generates a signature string for a Wowza Stream Cloud API request.
# @param String request_path The request path without the protocol or host name.
# @param String api_key The api key assigned to you through Wowza Streaming Cloud.
# @param Integer timestamp The Unix Epoch timestamp of the request

def generate_request_signature(request_path, api_key, timestamp)
  # Ensure we only have the path, and not the parameters
  request_path = request_path.split(?").first

  # Ensure the leading slash
  request_path = "/#{request_path}" unless request_path.start_with?("/")

  # Ensure the trailing slash is removed if it is present.
  request_path = request_path.chomp('/')

  # Generate the data string
  data = "#{timestamp}:#{request_path}:#{api_key}"

  # Return the HMAC-SHA256-Hex string.
  return OpenSSL::HMAC.hexdigest("SHA256", api_key, data)
end

Java example:
We pass the resulting value into the request header along with the timestamp and access key. The API request will look something like this:

```bash
curl -X GET \
-H "wsc-access-key: [key]" \
-H "wsc-timestamp: [timestamp]" \
-H "wsc-signature: [code-generated-signature]" \
"https://api.cloud.wowza.com/api/[version]/live_streams"
```

**API key and access key authentication**

Because using an account's API key and an access key directly in request headers is a less secure authentication method, we recommend it for initial evaluation and
proof of concept applications only. See Locate an API key and generate an access key for more information.

A request using the API key and access key authentication method looks like this:

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

**Important:** In example requests shown in our example articles, we use environment variables for the API key, access key, host, and version number. See Using cURL to learn more.

### Using the API sandbox

You can explore the Wowza Streaming Cloud REST API using the API sandbox. The API sandbox allows you to test and develop streaming workflows in a safe, separate environment that doesn't affect your Wowza Streaming Cloud production environment. The sandbox is fully functional and can do everything that the real API does, except that it creates simulated live streams and transcoders, not real ones. Because the sandbox doesn't use real resources, you don't accrue charges when you use it.

The Wowza Streaming Cloud REST API sandbox URL is

```
https://sandbox.cloud.wowza.com/
```

**Open the API sandbox.**

### Tools for testing the API

#### Using cURL

The Wowza Streaming Cloud REST API examples in this documentation site use curl commands. cURL is a command-line tool that allows you to execute HTTP
requests. It is native to the Terminal application on macOS and Linux, but it requires some installation for use in the Command Prompt on Windows operating systems. To find a download for Windows, see the curl Download Wizard.

To review curl command syntax used in our API examples, see API requests.

We use environment variables in the curl API request examples for Wowza Streaming Cloud to make it easier for you to copy, paste, and run commands in your Terminal or Command Prompt window. The variable syntax differs according to your operating system.

<table>
<thead>
<tr>
<th>Description</th>
<th>macOS/Linux Variable (used in examples)</th>
<th>Windows Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>API key</td>
<td>${WSC_API_KEY}</td>
<td>%WSC_API_KEY%</td>
</tr>
<tr>
<td>Access key</td>
<td>${WSC_ACCESS_KEY}</td>
<td>%WSC_ACCESS_KEY%</td>
</tr>
<tr>
<td>Host</td>
<td>${WSC_HOST}</td>
<td>%WSC_HOST%</td>
</tr>
<tr>
<td>API version</td>
<td>${WSC_VERSION}</td>
<td>%WSC_VERSION%</td>
</tr>
</tbody>
</table>

You can set environment variables like this, depending on your operating system:

Linux or macOS:

```bash
export WSC_ACCESS_KEY="your access key here"
export WSC_API_KEY="your API key here"
export WSC_HOST="https://api.sandbox.cloud.wowza.com"
export WSC_VERSION="v1.4"
```

Windows:

```bash
set WSC_ACCESS_KEY=your access key here
set WSC_API_KEY=your API key here
set WSC_HOST=https://api.sandbox.cloud.wowza.com
set WSC_VERSION=v1.4
```

After setting values for the environment variables, you can copy and paste API request examples into your Terminal or Command Prompt. You’ll still need to substitute resource IDs and update parameter values, where appropriate.
Using a GUI-based REST API client

Other options for testing the Wowza Streaming Cloud REST API are applications with a graphical user interface, such as Postman, Paw, or Insomnia, that provide a user interface and responses in formatted JSON. See documentation for your chosen API testing application for detailed usage information.

Note these configuration details:

<table>
<thead>
<tr>
<th>Authorization</th>
<th>No Auth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Headers</td>
<td></td>
</tr>
<tr>
<td>wsc-api-key:</td>
<td>[api key value for your account, for API key and access key authentication only]</td>
</tr>
<tr>
<td>wsc-access-key:</td>
<td>[access key value for your user]</td>
</tr>
<tr>
<td>wsc-timestamp:</td>
<td>[timestamp for the request, for HMAC authentication only]</td>
</tr>
<tr>
<td>wsc-signature:</td>
<td>[hash-encoded signature for the request, generated through a pre-request script or HMAC signature native to the API test application, for HMAC authentication only]</td>
</tr>
<tr>
<td>Content-Type:</td>
<td>application/json</td>
</tr>
<tr>
<td>Body</td>
<td>JSON-formatted request body that includes parameters and their assigned values</td>
</tr>
</tbody>
</table>

Example POST request in a GUI-based API test application:
Accessing support

Sometimes you need a helping hand. You can view the Wowza Streaming Cloud REST API community forum to get help from other Wowza Streaming Cloud users and experts from around the world.

You can also contact Wowza Support for REST API issues with Wowza Streaming Cloud. When contacting support for an error returned via the API, be sure to include the following information:

- The API request you sent that caused an error
- The request ID returned in the error response
- The request timestamp returned in the error response

The request ID and request timestamp will look something like this:
More resources

- Wowza Streaming Cloud REST API release notes
- Wowza Streaming Cloud REST API reference documentation (current version)

- Example application for the Wowza Streaming Cloud REST API, written in Ruby, on GitHub
- Get started broadcasting a live stream using the Wowza Streaming Cloud REST API
- 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
- Record a transcoder with the Wowza Streaming Cloud REST API
- Schedule a transcoder with the Wowza Streaming Cloud REST API
- View usage data with the Wowza Streaming Cloud REST API
- Create and view player URLs with the Wowza Streaming Cloud REST API