TV Conversions

Upload Data via API

iSpot.tv

Overview

This guide outlines the process and specifications for uploading TV Conversion data through the iSpot API. It is a secure and seamless way to integrate your impression data into the iSpot platform without the use of a tracking pixel.

Steps Required

There are many different methods you can utilize to upload your data through the API, but the general process will remain the same. We'll cover two methods below using Python and Curl.

1. Provide a sample file in the specified format to be approved by iSpot for upload.
2. Your Customer Success Representative will then provide your API credentials.
3. Gather the file(s) to upload and split/compress if necessary based on specifications below.
4. Authenticate using your API credentials.
5. Post your file(s) with passthru column (see "Available Dimensions" section)
6. Please notify your Customer Success Representative once successfully uploaded
7. Engineering will verify file contents meet the schema in "Available Dimensions"
8. If file contents meet schema requirements, files can be delivered daily, dropping the passthru column

CSV File Format

The file(s) must strictly adhere to the specifications outlined below in order to be processed.

- CSV UTF-8 (Comma delimited) (csv) - Rows are newline separated.
- .zip containing multiple files accepted.
- Compressed files must be under 200MB each, or the files won't be processed.

- Files must have unique file names for a given day. If two or more files with the same name are uploaded, only the first file will be kept
- Comma separated – All values (except for headers) enclosed in doublequotes.
- Headers are case-sensitive.

Example formatting of header row and one row of sample data:

```
ip,datetime,siteid,app,type,customdata,passthru,useragent
"66.87.152.32","2021-01-31
17:30:56","TC-####-#","web","purchase","customer_new,channel_paid-search","1","Mozilla/5.0
(Macintosh; Intel) AppleWebKit/537.36 (KHTML, like Gecko) Chrome 70.0.3359.117 Safari/537.36"
```

Upload Requirements

1. Files must be posted prior to 7:00 AM PDT/ 6:00 AM PST in order to be included in the standard daily processing.
2. Files should contain data for the previous day only and the event expressed as a UTC timestamp.
3. Data containing dates within the past 2 days OR data uploaded after 7:00 AM PDT/ 6:00 AM PST will be included in a recurring batch processing that occurs 12 days after the data date. This data will be available in the dashboard in 2 weeks.
4. Any data containing dates older than the past 12 days will have to be reprocessed incrementally and will incur a daily cost for each day that needs to be reprocessed. **Please align with your customer success representative prior to uploading data to set expectations.**

Matching Methodology

iSpot leverages the client's IP address to provide a wholistic measurement of all TV/OTT media and conversion metrics. While IPv4 is most commonplace, iSpot can support measurement of IPv6 addresses by integrating with a 3rd Party identity resolution partner. Use IPv4 if you want to match conversions to a Household.

Available Dimensions

While not all columns are required, it is recommended that you supply as many as possible for more robust measurement. The asterisks (*) below denote the dimensions that are currently available in the iSpot Analytics Platform.

Column Name	Description	Sample Value/Format	Required
ip	Client IPv4 Address	66.87.152.32	Yes
useragent	Client useragent	Mozilla/5.0 (Macintosh; Intel) AppleWebKit/537.36 (KHTML, like Gecko) Chrome 70.0.3359.117 Safari/537.36	Yes
datetime	Datetime of the event formatted as a 24 hr UTC timestamp	YYYY-MM-DD hh:mm:ss	Yes
siteid	Unique identifier *(Provided by Customer Success)*	TC-####-#	Yes
*app**	Type of device where the event happened *(values: "app" or "web")*	web	Yes
*type**	Description of a Web/App event or user action	Purchase	Yes

*customdata**	Custom defined details that add more granularity to the event "type"	customer_new,channel_paid-search	No
passthru	Static value 1 *(used only for validation purposes during testing)*	1	Yes

Other available Dimensions: (These dimensions are only available via custom reporting.)
orderid, amount, sku, customertype, channel

Authentication

The iSpot API utilizes oAuth2.0 for authentication, which involves using your client_id and client_secret to make a POST request to the authentication endpoint to retrieve a bearer token. This token is used for the subsequent API call in order to upload your data.

Credentials

Your credentials will be provided to you by your Customer Success representative and should be obtained prior to moving past this step. These credentials are for demonstration purposes only, but yours will be the same number of characters as outlined below.

Client ID

Your client_id is a 20-character alphanumeric value used to authenticate
Ex: 89ab614c1732d98e123f

Client Secret

Your client_secret is a 40-character alphanumeric value

Ex: Tr1f3k6PzStGhQcWLuMAcKdXr4g4s6rUcHwyTSby

Grant Type

When authenticating the grant_type parameter will always be the string 'client_credentials'

Retrieving a Bearer Token

Using the given credentials, you will make a POST request to the endpoint https://api.ispot.tv/v4/oauth2/token and pass your client_id, client_secret and grant_type as a parameter in the request headers. Authentication will vary depending on which method you are using to interact with our API, but the mechanics of the request are largely the same.

Token Expiration

Please note that your token is valid for a period of **24 hours**, at which point you will need to retrieve a new token. It is recommended that you limit the amount of token requests per day and only request a new one when absolutely necessary.

Example using curl from command line

Authenticating

```
curl --request POST \
--url 'https://api.ispot.tv/v4/oauth2/token' \
--data 'client_id=CLIENT_ID&client_secret=CLIENT_SECRET&grant_type=client_credentials'
```

After running the previous command, it should return your Bearer Token as shown below:

Sample Response with Bearer Token

{"token_type":"Bearer","expires_in":86400,"access_token":"eyJ0eTAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJerWQiOiIyWeLwMPBkMDdjZjRmZWRiYjZkNSIsImp0aSI6IjUyNDgxY2YzZTVmRQTkOGFiZGI1YWQ3NWRhMjg4NDFlYjE3MTE2M2JmZGNiYjFkZWJhOTg1ZDRjZmUyZjY0ODExOTBlYjM5NWI1NWViODM4IiwiaWF0IjoiMTYxSjg3MTIxTR45OTU0NzMiLCJuYmYiOiIxNjE2ODcxMjExLjk5NTQ3NyIsImV4cCI6IjE2MTY5NTc2MTEuOTg2NDE5Iiwic3ViIjoiIiwic2NvcGVzIjpbXX0.jh0UPXZR6CgPWMCTRYUpQMk5AfBIlZJ3Vcw1Tlh1ZLeiAh3pZwAKsLVEt2mAvbC_XD77CWxx_1yub-O2TX2Ty4GrjZms3XQBJw_xE6A6LQ77XNZ5E3iX0xEFOvIAlKx7MPtOGlHLsZg0W-2cDepLyE-xUMIplp6hul2ndId12rZOcEQgtT44W7XIKUIFtpGZCGdjQf13AfHYIfT2elnTRgZifnNLjpAkW9mx1Qas5LXzLx6PTLUnclf3kCAwRSsTTxfmspKZDfl58I6LGxVW24p1Uw71RrdkJGzuuUdIpz25VTKxx_3ZSB86mEwzDABYalSGyqgXv0vi9eLoR7d_ug"}

Your bearer token is a 671-character alphanumeric value valid for 24 hours (86400 seconds)

Making an API call

The bearer token returned in the previous step will be used to make the subsequent API upload. You'll want to copy and paste or store your bearer token in a variable, which will be passed through the request headers as a parameter. *Ex: --header 'Authorization: Bearer TOKEN_HERE'*

Splitting your Files

If your file(s) are larger than 200 MB they must be split prior to uploading via the API. Splitting files can be tedious, which is why we've provided a couple different ways in this section that you can use to split them programmatically in seconds.

Python3

The Python file splitting method converts a single large csv file into multiple csv files that are each 200 MB in size. Simply specify the full path to the file as indicated in the following example, and then run it in any Python3 environment.

```
from itertools import chain

def split_file(filename, pattern, size):

with open('/Specify/fullPath/to/file.csv', 'rb') as f:

    header = next(f) for index, line in
    enumerate(f, start=100):
      with open(pattern.format(index), 'wb') as out:
        out.write(header) n
        = 0 for line in
        chain([line], f):
          out.write(lin
          e) n +=
          len(line) if n
          >= size:
          break
if __name__ == '__main__': split_file('data.csv',

'file{0:03d}.csv', 200000000) print('\nDone.\n')
```

Change path to your file name

Files break out at 200 MB specified (in bytes)

Bash

The Bash file splitting method converts a single large csv file into multiple csv files specified by the amount of lines. You will need to determine the appropriate number of lines to keep the file size under 200 MB. Run this method in a terminal in the folder where your csv file is located. Change the file name and the number of lines where indicated in green in the following example.

```
tail -n +2 test.csv | split -l 500000; for file in `ls xa*`; do head -n 1 test.csv > tmp_file; cat $file >> tmp_file; mv -f tmp_file $file; done; for file in xa*; do mv "$file" "$file.csv"; done
```

Upload File(s) via Curl

```
curl -X POST https://api.ispot.tv/v4/metrics/conversions -H 'Authorization: Bearer YOUR_TOKEN' -H 'content-type: multipart/form-data' -F 'upload[file]=@filename.csv'
```

Upload File(s) via Python

```
import http.client, json, csv, requests

def get_token():
   conn = http.client.HTTPSConnection("api.ispot.tv")

   client_id = 'YOUR_CLIENT_ID'
   client_secret = 'YOUR_CLIENT_SECRET'
   grant_type = 'client_credentials'

   payload =
"client_id={CLIENT_ID}&client_secret={SECRET}&grant_type={GRANT_TYPE}".format(CLIENT_ID=client_id,
SECRET=client_secret, GRANT_TYPE=grant_type)

   headers = {
      'content-type': "application/x-www-form-urlencoded"
   }

   conn.request("POST", "/v4/oauth2/token", payload, headers)

   res = conn.getresponse()
   token_raw = res.read()
   data_parsed = json.loads(token_raw.decode("utf-8"))

   access_token = data_parsed['access_token']
   conn.close()
   return (access_token)

file_name = 'filename.csv'
files = {'upload[file]': (file_name, open(file_name, 'rb'), "multipart/form-data"),}


conn = http.client.HTTPSConnection("api.ispot.tv")
payload = ""
headers = {
   'Authorization': "Bearer {TOKEN}".format(TOKEN=get_token()),
   }

try:
   response = requests.post('https://api.ispot.tv/v4/metrics/conversions', files=files, headers=headers )
   print(response.status_code, files, response.text)
except Exception as err:
```

The following table provides a list of iSpot API endpoint responses.

HTTP status	Message	Case
201 (Created)	Uploaded. Our team will process your request soon! Thank you.	The CSV file has been successful submitted.
400 (Bad Request)	The content type request is not valid.	The headers and the content in the request are not correct. You must provide multipart/form-data to successfully submit the CSV format file.
400 (Bad Request)	The request must contain a CSV format file to be imported.	The headers are present, but a CSV format file was not submitted with the request.
400 (Bad Request)	The file cannot be processed because it is not in the required CSV file format.	The file submitted is not in a CSV file format. Only correctly formatted CSV files are supported.
400 (Bad Request)	The content is not valid. Please validate the headers and content.	For CSV files, the first line of the file must contain the following mandatory headers: LATEST_IPADDRESS, SITE_ID, TIME_STAMP
400 (Bad Request)	The file is too small. The minimum file size allowed is 8 MB.	The size of the file submitted is smaller than the minimum size required.
400 (Bad Request)	The file is too large. The maximum file size allowed is 200 MB.	The size of the file submitted is larger than the maximum allowed.
401 (Unauthorized)	Unauthorized	The credentials submitted for either username or password are incorrect.
409 (Conflict)	The file already exists.	The CSV file submitted is already created and published on iSpot.
500 (Internal Error)	Internal error. Please contact our API team.	An unknown error occurred. Contact the API team.
500 (Internal Error)	The server disk is full. Please contact our API team.	Insufficient server space prevents the process from handling file requests.