Event Protect Platform

RESTfull API call

Introduction

Via available online service and through specified API, developers can connect to Event Protect platform and submit individual sales transaction. Service is available 24/7 and in order to have access it is mandatory to have Agent account at Event Protect platform. New Agents who do not have account can start registration process <here>.

Submitting transaction with REST API

The Event Protect API is a RESTful web service. Character encoding is UTF-16. For adding new transaction, a PUT/POST http method is used. Other methods will return error message.

Each transaction log entry has to have predefined set of attributes and must conform to the specified rules. Requests to service must include following:

Request URL & Method:

URL for **UAT** environment:

```
https://api-uat.protect-platform.com/transactionapi/v1/
transaction
```

URL for **Production** environment:

```
https://api.protect-platform.com/transactionapi/v1/transaction
```

Request Method:

- Accepted METHODs: **POST** or **PUT**

Request HEADER:

- Content-Type: application/json
- X-EventProtect-VendorId: <*unified vendor Id provided by the platform*>
- X-EventProtect-AuthToken: <*please refer to authentication section*>

Request CONTENT:

Data format is JSON, with the following structure:

```
{
  "vendorCode": "DemoVendorId",
  "vendorTransactionReferenceId": "UniqueID-9166-9877-1234-3211",

  "customerName": "John Doe",
  "value": 15.50,
  "currencyCode": "USD",
  "quantity": 1,
  "transactionDate": "2016-09-15T22:32:28.000084+02:00",

  "eventData": {
      "eventName": "Really Cool Event",
      "eventDate": "2017-07-01",
      "eventType": "MOT",
      "country": "USA",
      "state": "US-LA",
      "city": "New Orleans",
      "zip": "70122",
      "street": "6801 Franklin Ave",
      "venueName": "Lakefront Arena"
  }
}
```

Request timeout:

In case no response is returned from API, a request timeout should be set to 10 seconds.

Data structure

Data structure is separated into two entities:

- Transaction related data
- Event related data (part of transaction related data)

Transaction data

Data field	Mandatory	Data type	Description
vendorCode	**YES**	Character	Unified vendor Id provided by the platform.
vendorTransactionReferenceId	**YES**	Character	Unique transaction ID coming from the Vendor's system. The format allows any type of id (numeric/character). **The combination of vendor id & vendor transaction reference id MUST be UNIQUE.**
customerName	NO	Character	Contains the name of the buyer in any format sent from the vendor. The platform does not check this data, but it will be used as additional reference for individual transaction manipulation (i.e. in case of claims or in case of transaction cancellation).
value	**YES**	Number	The price of ticket(s) including fees and charges. This is the value that is going to be insured. It is accepted as numeric value with following format (#0.00): ▪ Decimal separator: ".". ▪ Number of digits: 2 ▪ Thousands separator: NO ▪ Leading zero: YES
currencyCode	**YES**	Character	The code of the currency. Standard ISO 4217 Currency code is accepted (three letter code). Reference for format: https://en.wikipedia.org/wiki/ISO_4217
quantity	**YES**	Number	Number of items in the transaction. It is accepted as integer value with following format (#0)
transactionDate	**YES**	Character	The date & time of the transaction. Sent to the platform in ISO 8601 UTC format (YYYY-MM-DDTHH:mi:SS.ssssss+01:00) with time zone. Reference for format: https://en.wikipedia.org/wiki/ISO_8601
eventData	**YES**	EVENT DATA*	Event data nested as eventData field.

*described in next section

Event data

Data field	Mandatory	Data type	Description
eventName	**YES**	Character	The name of the event. Value in any format is accepted.
eventDate	**YES**	Character	The date of the event. Sent to the platform in ISO 8601 DATE format (YYYY-MM-DD). Reference for format: https://en.wikipedia.org/wiki/ISO_8601
eventType	**YES**	Character	The type of the event in terms of event type code. Event type codes are defined within the platform. Event types are listed below in *Table 1* - event types.
country	**YES**	Character	The code of the country where event is taking place. Standard ISO 3166-1 ALPHA 3 Country code is accepted. Reference for format: https://en.wikipedia.org/wiki/ISO_3166-1
state	NO	Character	The state of the event; i.e. administrative level 1 division within a country - state, province, territory). Use Country subdivision code as defined by ISO 3166-2. Sent data must comply to this format. Reference for format: https://en.wikipedia.org/wiki/ISO_3166-2
city	**YES**	Character	The city where event is taking place. The platform does not check validity of this data.
zip	NO	Character	ZIP code of the event location. The platform does not check validity of this data, but it will be used (if supplied) to uniquely define an event and its geolocation (coarse).
street	NO	Character	Full address of place where event is taking place. The platform does not check this data but it is recommended to specify full address of event location (street, street number, zip code, ...)
venueName	NO	Character	The name of the venue the event is taking place.

Event types

Below are listed allowed event types.

Event Type	Event Type	Event Type
PIN	Professional Indoor	Professional Indoor - This is a conference, show or event that is for professionals and/or the general public. The event will be held in an indoor venue such as a conference centre or hotel.
POT	Professional Outdoor	Professional Outdoor - This is a conference, show or event that is for professionals and/or the general public. The event will be held at an outdoor venue such as a field or inside an outdoor temporary structure such as tent or marquee.
MIN	Music Indoor	Music Indoor - A music lead or based performance, which could include Festival/s or Concert/s. This type of event will be held in an indoor theatre or venue.
MOT	Music Outdoor	Music Outdoor - A music lead or based performance, which could include Festival/s or Concert/s. The event will be held at an outdoor venue such as a field or inside an outdoor temporary structure such as tent or marquee.
SIN	Sports Indoor	Sport Indoor - A sports based spectator event such as ice hockey, indoor basketball, etc. This type of event will be held in an indoor venue such as ice rink, forum or stadium which is completely covered and not subject to cancellation due to adverse weather.
SOT	Sports Outdoor	Sport Outdoor - A sports based spectator event such as soccer, cricket, outdoor tennis, etc. The event will be held at an outdoor venue such as a field or an outdoor stadium.
AIN	Activity Indoor	Activity Indoor - An event that requires some form of participation or activity by the consumers, examples would be climbing centres, go-karting tracks, activity centres, etc. These events are normally not impacted by adverse weather and are completely indoor.
AOT	Activity Outdoor	Activity Outdoor - An event that requires some form of participation or activity by the consumers, examples would be triathlons, mud runs, etc. The event will be held at an outdoor venue such as a field or inside an outdoor temporary structure such as tent or marquee.

GIN	General Indoor	General Indoor - Any other event type that does not more specifically fit within the with the event type definitions above. These events are normally not impacted by any adverse weather and are completely indoor.
GOT	General Outdoor	General Outdoor - Any other event type that does not more specifically fit within the with the event type definitions above. The event will be held at an outdoor venue such as a field or inside an outdoor temporary structure such as tent or marquee.
COT	Coastal Outdoor	Coastal Outdoor - Any outdoor event that is located less than <1km from a sea, ocean, lake, river or other large body of open water

Authentication

In order to submit transaction each method call needs to be authenticated. Authentication method to sign & authenticate REST requests from 3rd party applications is HMAC-SHA256. For authentication process it is necessary to have authentication token which is sent in request HEADER in field “X-EventProtect-AuthToken”. Authentication token is created by combining two values

- **vendorID**
  Unified vendor Id provided by the platform
- **current date**
  UTC date on which data is sent in ISO 8601 DATE format (YYYY-MM-DD). For example, 15th of June 2016 would be 2016-06-15

Once these two values are combined, HMAC-SHA256 digest of this value is generated using your API Key.

Finally, this value must be base64 encoded.
Both **vendorID** and the **API key** are provided by the platform, and are available under members’ access.

Code examples for AuthToken

C#

```
1.  string vendorID = "DemoVendorId";
2.  string APIKey = "DemoAPIKey";
3.
4.  var encoding = new ASCIIEncoding();
5.  byte[] keyByte = encoding.GetBytes(APIKey);
6.  byte[] messageBytes = encoding.GetBytes(vendorID + DateTime.UtcNow.ToString("yyyy-MM-
    dd"));
7.  using (var hmacsha256 = new HMACSHA256(keyByte))
8.  {
9.      byte[] hashmessage = hmacsha256.ComputeHash(messageBytes);
10.     return Convert.ToBase64String(hashmessage);
11. }
12.
```

Java

```java
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.text.SimpleDateFormat;
import org.apache.commons.codec.binary.Base64;


String vendor = "DemoVendorId";
String key = "DemoAPIKey";

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
sdf.setTimeZone(new SimpleTimeZone(SimpleTimeZone.UTC_TIME, "UTC"));

String tstamp = sdf.format(new Date());
String message = vendor + tstamp;

Mac sha256_HMAC = Mac.getInstance("HmacSHA256");
sha256_HMAC.init(new SecretKeySpec(key.getBytes(), "HmacSHA256"));

String authToken = new String
(Base64.encodeBase64(sha256_HMAC.doFinal(message.getBytes())));
```

PHP

```php
$vendorId = "DemoVendorId";
$key = "DemoAPIKey";

$date = gmdate("Y-m-d");
$message = $vendorId . $date;

$hash = hash_hmac('sha256', $message, $key);
$hash = pack('H*', $hash);

$authTokenB64 = base64_encode($hash);
```

Python

```python
import base64
import hmac
from hashlib import sha256
from datetime import datetime

VENDOR_ID = 'DemoVendorId'
API_KEY = 'DemoAPIKey'


def gen_base64_digest_str(message, key):
    hmac_gen = hmac.new(bytes(key, 'ascii'), digestmod=sha256)
    hmac_gen.update(bytes(message, 'ascii'))

    return base64.b64encode(hmac_gen.digest()).decode('ascii')


def create_token():
    utcDateISO = datetime.utcnow().strftime('%Y-%m-%d')
    message = VENDOR_ID + utcDateISO

    return gen_base64_digest_str(message, API_KEY)


print 'Valid token: ' + create_token()
```

AuthToken examples

Date	Vendor ID	Message	Key	AuthToken
2016-10-05	DemoVendorId	DemoVendorId2016-10-05	DemoAPIKey	TsfF7o5c1OxcEZkwH47R16+P7pQRpGYo7cEBebdFrRU=
2016-10-06	DemoVendorId	DemoVendorId2016-10-06	DemoAPIKey	FUTYDacR1XH+agt0fGNSrzk+MLwq4r5hUVCJMEQVWrY=

Response signals

Depending on input parameters and service status following response codes are possible:

Code	Text	
200	OK	OK signal
400	Bad Request	Non implemented URI call
401	Unauthorized	Missing authorization
405	Method Not Allowed	Non PUT/POST method
408	Request Timeout	Long response
413	Request Entity Too Large	Submitted entity too large
422	Non-processable Entity	Missing data (data submitted not ok; e.g. missing price)
500	Internal Server Error	Internal error occurred. No data stored.
507	Insufficient Storage	In case no data

Validation of requests

Before you get credentials, developers can start coding the connection to Event Protect platform.

For that purpose, the TEST API service has been created. It is a different URL, but offers the same functionality as a regular API, just no data is saved. This service will return one of the response values, depending on submitted data.

To use test service, it is necessary to use test URL and demo credential data, so header call will have following parameters.

Request URL & Method:

- URL for **UAT** environment:
    - `https://api-uat.protect-platform.com/transactionapi/v1/transactiontest`

- URL for **Production** environment:
    - `https://api.protect-platform.com/transactionapi/v1/transactiontest`
- Accepted METHOD:
    - **POST** or **PUT**
- Test Vendor ID:
    - DemoVendorId
- Test Vendor API key:
    - DemoAPIKey

For test service, you can also use your own Vendor ID & API key provided to you by the platform.

Request example

```
POST /transactionapi/v1/transactiontest HTTP/1.1
Host: api-uat.protect-platform.com
Content-Type: application/json; charset=utf-8
X-EventProtect-VendorId: DemoVendorId
X-EventProtect-AuthToken:
FUTYDacR1XH+agt0fGNSrzk+MLwq4r5hUVCJMEQVWrY=
Cache-Control: no-cache

{
  "vendorCode": "DemoVendorId",
  "vendorTransactionReferenceId": "UniqueID-9166-9877-1234-3211",

  "customerName": "John Doe",
  "value": 15.50,
  "currencyCode": "USD",
  "quantity": 1,
  "transactionDate": "2016-10-06T20:32:28.000084+02:00",

  "eventData": {
      "eventName": "Really Cool Event",
      "eventDate": "2017-07-01",
      "eventType": "MOT",
      "country": "USA",
      "state": "US-LA",
      "city": "New Orleans",
      "zip": "70122",
      "street": "6801 Franklin Ave",
      "venueName": "Lakefront Arena"
  }
}
```

Suggested development process

1. Request your Vendor Id and API key from the platform manager.
2. Develop your code and test it using platform test/demo functionality. Use DEMO Vendor Id and Demo Key.

- DEMO Vendor Id: DemoVendorId
- DEMO Vendor Key: DemoAPIKey
- Make calls to: https://api-uat.protect-platform.com/transactionapi/v1/transactiontest
- Use one of code snippets from this document to make authentication token.
- Use json data from this document as a content to start with.
- Use your own data to create API content and validate it.

3. Once credentials have been provided by the platform (Vendor Id, Vendor API Key), test your code with new credentials.
4. Move to UAT:
    - Switch calls to: https://api-uat.protect-platform.com/transactionapi/v1/transaction
    - Check if your transactions are visible in the platform (leave up to 1hr for submitted transactions to propagate to the platform).
5. Once approved, move to production Environment.
    - Test it: https://api.protect-platform.com/transactionapi/v1/transactiontest
    - Move to production: https://api.protect-platform.com/transactionapi/v1/transaction