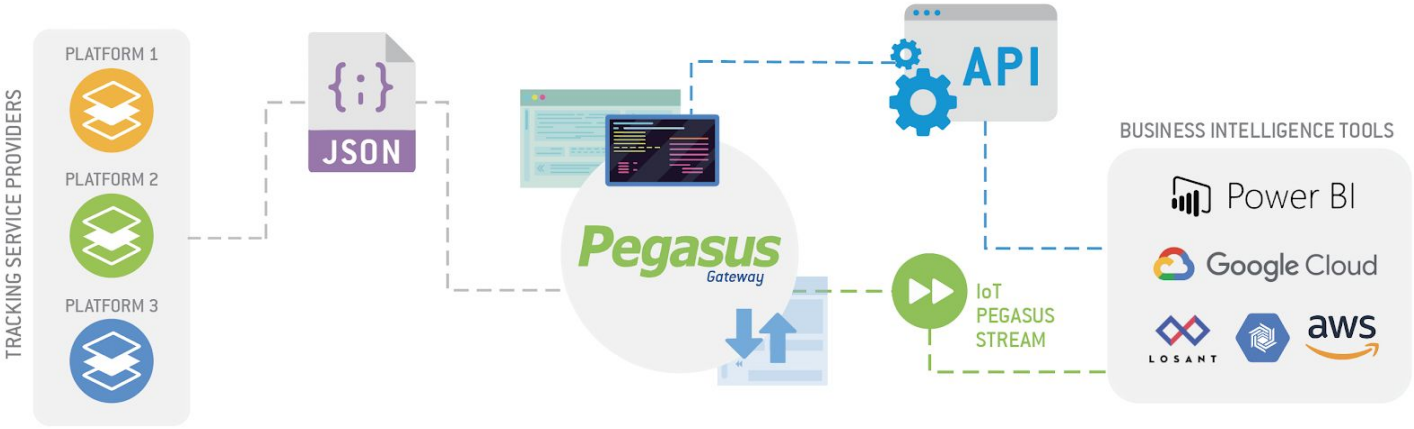
Pegasus Interoperability Data Receiver

v1.4 - JUL/2019



Introduction

This document describes the steps to get started in uploading telemetry data from other platforms to Pegasus servers. Data is uploaded in key-value pairs in a JSON array format via an HTTP POST request.

```
[{"boolean":true, "number":73, "string":"value"}]
```

Arrays of up to 50 JSON objects can be sent simultaneously. Each object corresponds to an event generated by different telemetry devices which has locational data.

```
[{"event1":"value1"}, {"event2":"value2"}, ... {"event49":"value49"}]
```

API Description

Endpoint: **RECEIVER_FROM_ADMIN_STEP_1**

Example: https://pegasusXX.peginstances.com/receivers/json

Method: **POST**

Headers: **Content-Type: application/json**

Body: **ARRAY_OF_EVENTS**

Requirements

The required parameters to send in the **ARRAY_OF_EVENTS** are:

- **`device.id`** - IMEI of the device
- **`position.latitude`** - latitude (example: 20.99385)
- **`position.longitude`** - longitude (example: -89.71079)

If these fields are not received the message is considered a keep alive, and the contents are discarded.

The other fields in the payload are not required but can be very useful, they are described below.

Payload Parameters

PARAM	TYPE	DESCRIPTION	REQUIRED
timestamp	number	Unix epoch timestamp (if not provided a server timestamp will be applied to the incoming message time)	no
device.name	string	Name for the device	no
device.id	number	IMEI on Pegasus (15 digit)	yes
position.latitude	number	WGS84 latitude	yes
position.longitude	number	WGS84 longitude	yes
position.direction	number	Heading in degrees 0 - 359 (0 = North, 90 = East, etc.)	no
position.speed	number	Speed in kph	no
position.altitude	number	Altitude in meters	no
position.hdop	number	Horizontal dilution of precision (Wiki)	no
position.satellites	number	Amount of GPS Satellites	no
position.valid	boolean	True if the positional data is valid	no
event.enum	number	Custom numeric code	no
event.label	string	Short string that describes the payload content, info	no
io.ignition	boolean	True if ignition is detected	no
io.power	boolean	True if main power is detected	no
io.input1	boolean	True if a device input is detected	no
device.battery.level	number	Battery level in milli-volts	no
device.battery.percent	number	Battery level as a percent	no
metric.odometer	number	Odometer in km (decimals accepted)	no
metric.hourmeter	number	Engine hours (decimals accepted)	no

Payload Example

```
[
        {
          "timestamp": 1561475702,
          "device.name": "MyDevice12345",
          "device.id": 350000000012345,
          "position.latitude": 10.356101,
          "position.longitude": -75.495798,
          "position.direction": 245,
          "position.speed": 20,
          "position.altitude": 13,
          "position.hdop": 0.6,
          "position.satellites": 12,
          "position.valid": true,
          "event.enum": 24,
          "event.label": "ignon",
          "io.ignition": true,
          "io.power": true,
          "io.input1": true,
          "device.battery.level": 3420,
          "metric.odometer": 66675.456,
          "metric.hourmeter": 375.123
        }
]
```

Response

The expected response from the receiver is HTTP status 200, “ok” when the data was successfully uploaded.

A wrong payload returns a 400 Bad Request.

Things to Consider

- The io params (`io.ignition`, `io.power`, `io.input1`) persist until they're sent as false.
- Sending the `valid_position` parameter as true overrides the internal filters that calculate the validity of the position using other parameters, send it as true **only** if you are sure that it's valid.
- If the epoch `timestamp` is not provided, as soon as the message is received the server will apply a Unix timestamp as the time of the event.
- Whenever you send the timestamp make sure to consider sending them in chronological order whenever the device comes back online after being offline. It's not required, but it's strongly recommended.
- Input 1 behaves as both a normal two state ON/OFF switch, or a panic alert depending on the label that's associated to the payload.
- A valid 15 digit IMEI is required on the `device.id`

Labels

Labels are short strings that represent the action that took place on any device, they are meant to give meaning/significance to the messages a telemetry device sends. For example if the Ignition cable on the device was detected ON, the payload of the event can report the label `ignon`. This tells the API that the message received was the ignition ON event. If the device was going over the speed limit, it will use the label `spd`, and so on.

LABEL	DESCRIPTION
`trckpnt`	Periodic report with Ignition ON (default if no label is set)
`prdtst`	Periodic report with ignition OFF
`ignon`	Ignition was turned ON
`ignoff`	Ignition was turned OFF
`panic`	Panic button was pressed
`pwrloss`	Main power was lost
`pwrrstd`	Main power was restored
`lwbatt`	Low internal device battery
`spd`	Speeding event
`idl`	Idling event
`stt`	Device motion detected
`stp`	Device stopped moving
`in1on`	Input 1 ON
`in1off`	Input 1 OFF

Changelogs

v1.0 - JUN/2019 - First release

v1.1 - JUN/2019 - Added 15 digit IMEI requirement

v1.2 - JUN/2019 - Updated event.code to event.enum

v1.3 - JUL/2019 - Changed example to an array

v1.4 - JUL/2019 - Added image