ODAMS RESTful API v1.0 Documentation

Table of Contents

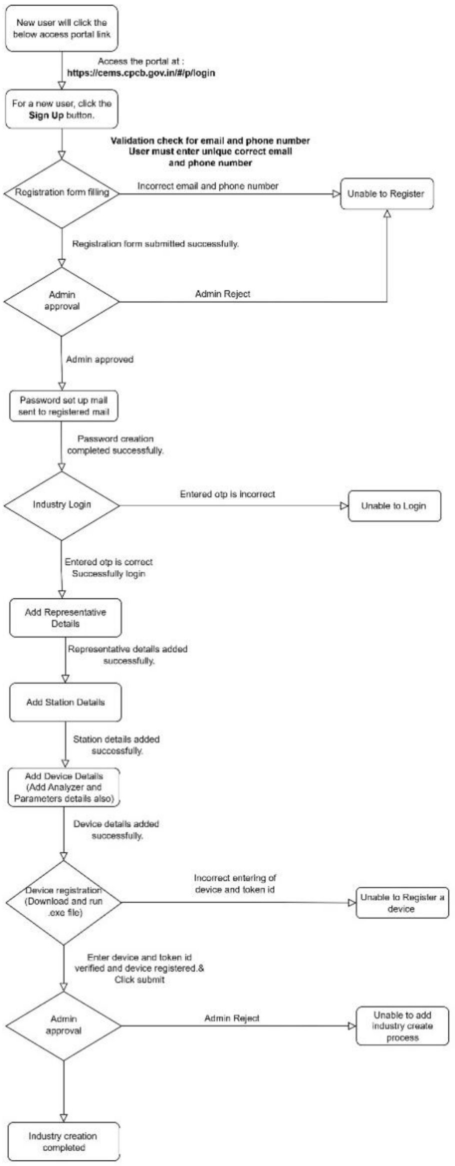
Document Version: 1.5

1. Overview

The **OCEMS REST API** requires all payloads to be transmitted in an **encrypted format**, which must be generated using a designated programming script. Both the **payload content** and the **HTTP headers** are expected to be provided in their expected form as defined by the API specifications.

Technology Reference: For ease of integration and to ensure uniform implementation of encryption and signature generation, the **sample implementation provided in this document uses Python** as the reference programming language.

1.1 — Flow diagram to understand the Steps of integration for the users.



New user will click the below access portal link
Access the portal at : https://cems.cpcb.gov.in/#/p/login
For a new user, click the Sign Up button.
Validation check for email and phone number User must enter unique correct email and phone number
Registration form filling
Incorrect email and phone number
Unable to Register
Registration form submitted successfully.
Admin approval
Admin Reject
Admin approved
Password set up mail sent to registered mail
Password creation completed successfully.
Industry Login
Entered otp is incorrect
Unable to Login
Entered otp is correct Successfully login
Add Representative Details
Representative details added successfully.
Add Station Details
Station details added successfully.
Add Device Details (Add Analyzer and Parameters details also)
Device details added successfully.
Device registration (Download and run .exe file)
Incorrect entering of device and token id
Unable to Register a device
Enter device and token id verified and device registered.& Click submit
Admin approval
Admin Reject
Unable to add industry create process
Industry creation completed

2. API to Upload Data of an Industry

This API is used to upload parameter values of multiple stations of an industry. A station can be an ETP (Effluent Treatment Plant) or a stack. A station can have multiple analyzers installed to record different environmental parameters (COD, BOD, TSS, pH, flow, SOx, NOx, PM, etc.).

2.1 Endpoint

```
https://cems.cpcb.gov.in/v1.0/industry/data
```

2.2 Method

```
POST
```

2.3 Headers

The request header should contain:

Header Key	Description	Value
signature	AES encrypted key	Base64 encoded encrypted signature (see generation steps below)
X-Device-Id	Unique device identifier	Device ID received in email after registration of device; in case Device ID is not received, please drop a mail to cems.cpcb@nic.in

2.3.1 IOT ID

The IOT ID received during registration must be included in the request header using the key X-Device-Id.

```
X-Device-Id: <device_id_from_registration>
```

Important Notes:

- **Each station can have multiple IoT devices or IOT IDs associated with it. But 1 IOT ID cannot have multiple Stations.**

2.3.2 Signature Generation

The signature header value is generated through a multi-step encryption process that combines a token ID with the current timestamp (YYYY-MM-DD HH:MM:SS.mmm).

Signature Key Generation: Encrypts a string that combines a `token_id` and the current timestamp using an RSA public key and OAEP padding with SHA-256. with a separator '$*' in between token Id and the current timestamp.

Step by Step execution:

1. **Data to encrypt :** Data must be encrypted each time with Token ID and a current timestamp prefixed with "$*" (Dollar followed by *)

```
message = token_id + str("$*" + str(datetime.now()))
```

2. **Padding Scheme (OAEP with SHA-256) :** The RSA encryption in this API uses OAEP padding with SHA-256.

```
mgf=padding.MGF1(algorithm=hashes.SHA256()),
algorithm=hashes.SHA256(),
label=None
```

3. **Encrypting with Public Key :** Message generated during first step and padding from 2nd step to be used for encryption.

```
public_key.encrypt(message, padding)
```

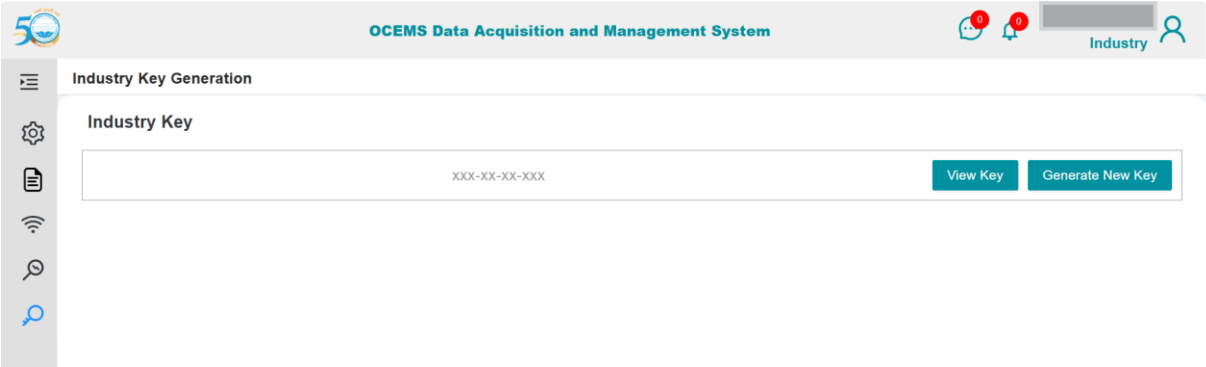
4. **Signature - Base64 Encoding:** The encrypted payload in step3 to be converted into a Base64 string with ‘utf-8’

2.3.3 Header example :

```
signature:<base64_encoded_encrypted_signature>
X-Device-Id: <device_id_from_registration>
```

Important Notes:

- **Public Key will be generated from the key generation page, and the same details will be sent to registered email id.**
- **The above encryption must be done for each IOT device and during each time data is sent to CPCB server.**



OCEMS Data Acquisition and Management System

[redacted]

Industry Key Generation

Industry Key

XXX-XX-XX-XXX

2.4 Request Payload

The payload must be **AES encrypted** using:

- **Secret Key**: Token ID
- **Mode**: ECB
- **Algorithm**: SHA256

Payload will be encrypted using AES encryption with help of token id as secret key and mode in ECB and algorithm will be SHA256

Step by step execution:

1. **Derive the AES Key from Token Id:** Token id must be hashed with SHA-256 which produces binary value that becomes the AES key. Ensure the key is always the correct size (256 bits) for AES. As a result this will generate the hashed key.

2. **Create AES Cipher:** An AES cipher object must be created using the hashed key generated in step 1, with mode as ECB where each block is encrypted.

3. **Encrypt the payload:** Using Cipher encrypt the payload with padding to match AES’s block size of 16 bytes. As a result the cipher then encrypts this padded payload into binary ciphertext.

4. **Base64 Encoding:** The encrypted payload in step3 to be converted into a Base64 string with ‘utf-8’

5. The final string generated in step 4 to be used as a body(payload) content while transmitting the data to end point.

Important Notes:
- **Token Id will be generated during device registration page, and the same details will be sent to registered email id.**
- **The above encryption must be done for each IOT device.**

2.4.1 Example

Auth Token: `8d97c71e68c2f4eee5fb254fccb49773`

Sample JSON (before encryption):

```
{
 "data": [
   {
    "stationId": "xxxxxx",
    "device_data": [
      {
       "deviceId": " xxxxxx ",
       "params": [
         {
          "parameter": " xxxxxx ",
          "value": xxxxxx,
          "unit": " xxxxxx ",
          "timestamp": xxxxxx,
          "flag": "U|C|M|F|Z|D"
         }
       ],
       "diag_params": [
         {
          "parameter": "acid_mist",
          "value": xxxxxx,
          "unit": "mg2",
          "timestamp": xxxxxx,
          "flag": "U"
         }
       ]
      }
    ],
    "latitude": xxxxxx
    "longitude":  xxxxxx
   }
 ]
}
```

2.4.3 Data Field Description and Date Validation

Field Name	Type	Mandatory	Description	Example Value	Validation Rules
stationId	String	Yes	Unique ID of the station (ETP or Stack)	"station_2"	Must be a valid registered station ID.
deviceId	String	Yes	Unique ID of the IoT device assigned to the station	"device_1"	Must match registered device for the station.
parameter	String	Yes	Parameter key as per Annexure 3.1	"cod", "ph"	Must be one of the allowed keys.

Field Name	Type	Mandatory	Description	Example Value	Validation Rules
value	Number	Yes	Measured value for the parameter	245.5	Numeric only.
unit	String	Yes	Unit of the parameter	"mg/l"	Must match allowed unit for the parameter.
timestamp	Number (ms)	Yes	Unix Epoch timestamp in milliseconds (UTC or IST as specified)	1726838400000	- Must not be a future date - Must not be older than 7 days - Must align to 15-min slots
flag	String	Yes	Operation mode flag	"U", "C", "M", etc.	U = Normal, C = Calibration, etc.
latitude	Decimal	Optional	Latitude of station location	28.6129	Range: -90 to 90
longitude	Decimal	Optional	Longitude of station location	77.2295	Range: -180 to 180

Date & Time Validation Rules

- **Format:** UNIX Epoch time in milliseconds
- **Timezone:** IST (UTC +5:30)
- **Backdate limit:** Data older than 7 days is not accepted (Error 117)
- **Future date:** Not allowed (Error 118)
- **Interval rule:**
  - 1 record per 15-minute interval (00, 15, 30, 45)
  - During calibration mode, push data every 30 seconds
- **Example Valid Timestamps:**
  - 1726838400000 → 21-09-2024 00:00:00 IST
  - 1726839300000 → 21-09-2024 00:15:00 IST

Encrypted Payload:

```
ZOYAk5D9QsU3Ut+5ZX3ydHYjdSWy1/u43AQYva0X71XOnzJinFtX4fX7p5gtGsuV9vJC2iK8I2YmAwN9lO5Gb24za2+1ult4J
STq8+efkjUQ0FJ06KJRZDK5OE+qa8o9Fq7cRH37FP11cnm+azFhVoGTm9pJtTymD+gnKVE9mQcBMyHz0kxZ3Z2NCBZHc85rsN
aNPdKON5o3uOx+zYIvt8UFlNx/vtOW69SXd6rAg6dt95abhPSgkye4nV+dSSFbEE0C2pN04IShIkixGjqLBA==
```

2.4.3 JSON Structure

Parameter	Data Type	Description
`stationId`	String	ID of the station (ETP or Stack)
`data`	Array	the data object contains information related to each station with list of device details and the actual parameter name, datapoint value and timestamp associated to that device.

Important Notes:

- **When device is in Calibration mode or Zero Calibration mode, vendors should capture and push one data point every 30 seconds**
- **During operation, the time gap between two data points for the same parameter must be 15 minutes. This 15-minute average must be uploaded exactly at fixed 15-minute intervals (e.g., 00, 15, 30, 45 minutes).**

2.5 Response Codes

2.5.1 API Success Response

```
{"msg": "success", "status": 1}
```

2.5.2 API Error Responses

```
{
    "status": 10,
    "msg": "failed"
},

{
    "status": 102,
    "msg": "Invalid_Station"
},

{
    "status": 109,
    "msg": "Payload not encrypted properly"
},

{
    "status": 110,
    "msg": "Invalid unit"
},

{
    "status": 111,
    "msg": "Uploaded data is not matching with defined 15 min timeframe"
},
{
    "status": 112,
    "msg": "No calibration scheduled for this timestamp please contact cpcb"
},

{
    "status": 113,
    "msg": "signature key is missing in headers"
},

{
    "status": 114,
    "msg": "X-Device-Id key is missing in headers"
},

{
    "status": 115,
    "msg": "Public_Key is missing Generate the Key"
},

{
    "status": 116,
    "msg": " Device is not registered, Please register for the Industry "
},

{
    "status": 117,
    "msg": "Data cannot be pushed beyond 7 days"
},

{
    "status": 118,
    "msg": "Data cannot be pushed for future time"
},

{
    "status": 119,
    "msg": "Invalid Parameter"
}
{
    "status": 120,
    "msg": "Multiple Station Found in the Payload"
}
{
    "status": 121,
    "msg": "The Station and Device Mapping not Found in the Payload"
}
```

Note: Please refer to 3.3 for Error codes.

3. Annexure

3.1 Parameter Keys

Use these keys when submitting data for parameters:

Parameter Name	**Parameter Key**

Acid Mist	acid_mist
Ammonia	ammonia
Ammoniacal Nitrogen Concentration	ammonical_nitrogen
Ammonium	ammonium
an	an
AOx Concertration	aox
Arsenic Concertration	arsenic
Arsenic	as
Benzene	benzene
BOD	bod
Chlorine(Effluent)	chlorine
Chromium Concentration	chromium
CL	cl
CL2	cl2
CO	co
CO2	co2

Phosgene(COCl2)	cocl2
COD	cod
Color	color
Conductivity	conductivity
CR6	cr6
CS2	cs2
Cyanide	cyanide
Cynide Concentration	cynide
Dissolved Oxygen	do
Electrical Conductivity	ec
Flow Back Water	flow back water
Emission Flow	flow_emission
Flow Inlet	flow_inlet
Flow_inlet_totalizer	flow_inlet_totalizer
Flow Volume	flow_totalizer

Fluoride Concentration	fluoride
Fluoride Concentration	fluoride_effluent
Gas Flow	gas
H2O	h2o
H2S	h2s
HC	hc
HCL	hcl
HCN	hcn
HF	hf
HG	hg
Inlet FLow2	inlet flow2
Inlet Flow2 Totalizer	inlet flow2 totalizer
N03-N	n03-n
NH3-N	Nh3-n
NH3	nh3_effluent
NH4	nh4

NH4-N	nh4-n
Nickle Concentration	ni
NO3	no3
Nitrate as Nitrogen	no3-n
O2	o2
O3	o3
Oil-in-Water	oil in water
Oil & Grease	oil_grease

Parameter Name	**Parameter Key**
Opacity	opacity
pH	ph
Phenol Concentration	phenol
Phosphates Concentration	phosphates
Phosphorous	phosphorous
PM	pm
Pressure	pressure
PRIMARY TEMPERATURE	primary_temperature
Sec. Temp	secondary_temperature
SO2	so2
SO2 PPM	so2_ppm
SOX	sox
Sulphide	sulphide
TC	tc
TDS	tds

temp_test	temp_test
Env Temp	temperature
Stack Temperature	temperature_emission
THC	thc
TN	tn
TOC	toc
Toc	toc_effluent
THC	total hydrocarbon content(thc)
Total Chromium	total_chromium
Total Nitrogen	total_nitrogen
Totalizer	totalizer
TSS	tss
TVOC	tvoc
VCM	vcm
Velocity	velocity
VOC	voc

3.2 Unit Keys

Use these keys when submitting unit data:

Unit	Unit Key
%	%
°C	°C
°F	°F
cm	cm
cm/s	cm/s
cm^3	cm^3
F/m	F/m
g/m^3	g/m^3
H/m	H/m
Hrs	Hrs
Hz	Hz
Imperial gpm	Imperial gpm
inch	inch
K	K
Kg/Hr	Kg/Hr
kg/m^3	kg/m^3

km/hr	km/hr
kmph	kmph
knm3/h	knm3/h
L	L
Lat-Lng	Lat-Lng
l/hr	l/hr
l/min	l/min
l/s	l/s
m	m
m/s	m/s
m^2	m^2
m^3	m^3
m^3/day	m^3/day
m^3/hr	m^3/hr
m^3/s	m^3/s

mbar	mbar
mg/L	mg/L
mg/m³	mg/m³
mg/Nm³	mg/Nm³
mile	mile
miles/hr	miles/hr
min	min
mm	mm
Mole	Mole
mph	mph
mS/cm	mS/cm
Number	Number
Pa	Pa
pH	pH

ppb	ppb
ppm	ppm
ratio	ratio
s	s
T/D	T/D
Text	Text
ton/hr	ton/hr
TR	TR
True False	True False
uS/cm	uS/cm
US gpm	US gpm
µg/m³	µg/m³

3.3 Common Status Codes

Status Code	Description
1	Success
0	Request failed (unknown reason)
10	Invalid details under station and device
102	Invalid Station (The specified stationId or station is not created)
109	Payload not encrypted properly
110	Invalid Unit (The specified unit for the device is invalid or not added)
111	Uploaded data is not matching with defined 15 min timeframe.
112	No calibration scheduled for this timestamp please contact CPCB. (There is no scheduled calibration at the given time. Please contact CPCB for assistance)
113	Signature key is missing in headers
114	X-Device-Id key is missing in headers
115	Public_Key is missing Generate the Key
116	Device is not registered, Please register for the Industry
117	Data cannot be pushed beyond last 7 days
118	Data cannot be pushed for future time

119	Invalid Parameter
120	Multiple Station Found in the Payload
121	The Station and Device Mapping not Found in the Payload