

Accessing IQ Gateway local APIs or local UI with token-based authentication

Table of contents

Accessing IQ Gateway local APIs or local UI with token-based authentication	1
Generating the token	2
Obtaining a token via web UI.....	2
Obtaining a token programmatically via GET on a URL.....	4
Accessing the IQ Gateway using a token	6
Access the IQ Gateway local UI	6
Access the IQ Gateway APIs.....	7
Appendix	8
Getting meter details.....	8
Getting meter readings	9
Getting reported inverter production data	14
Getting meter’s live data	15
Getting power consumption data	18

At Enphase, we create high-quality solutions that meet the highest security standards. Several of our installers and homeowners use local APIs or local UI on the IQ Gateway to access data. These interfaces were in the past protected by conventional password-based authentication. With IQ Gateway software version 7.0.x or higher, local UI and APIs need cryptographic token-based authentication to improve security. This technical brief explains:

- How to obtain a token for your IQ Gateway
- How to access IQ Gateway local UI and APIs using the token

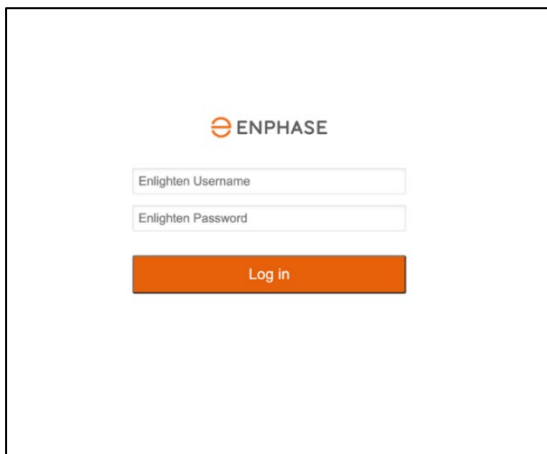
Generating the token

This section describes methods to generate a unique token that can then be used to access IQ Gateway local interfaces.

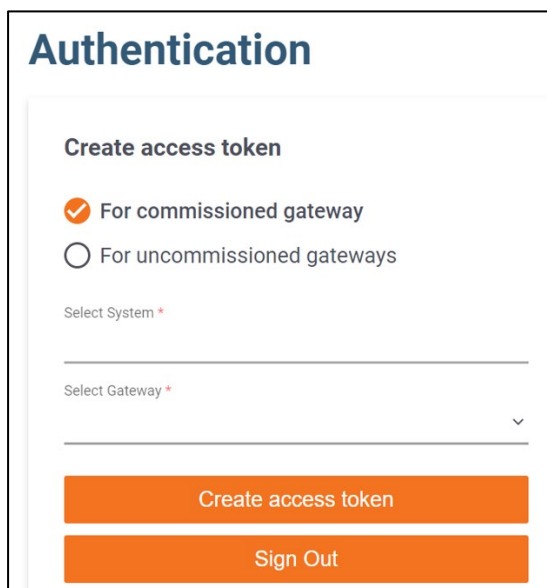
Obtaining a token via web UI

This method is useful when the user requires a one-time use token or does not require a programmatic way of token generation.

1. In the browser address window, enter <https://entrez.enphaseenergy.com>.
2. Click “Login” and enter your Enphase App credentials.

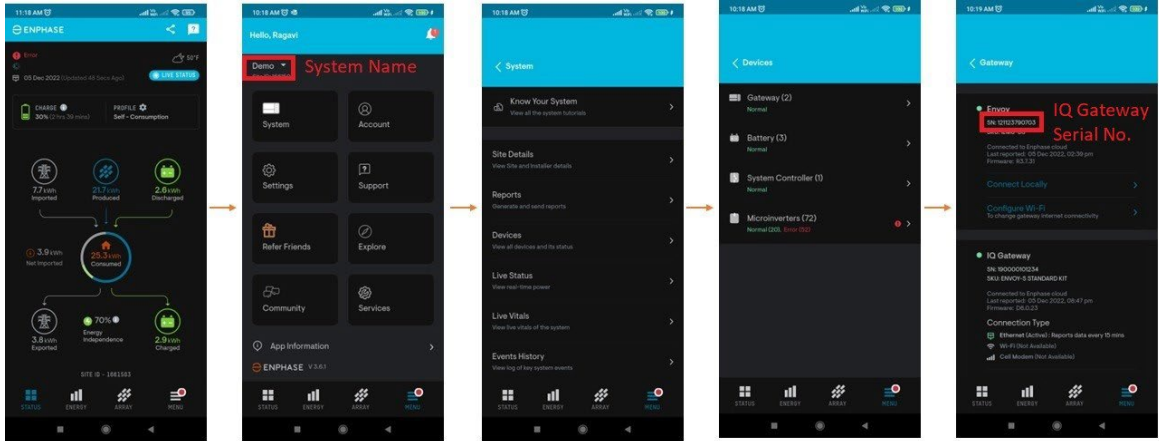


3. Select your system name and IQ Gateway serial number.



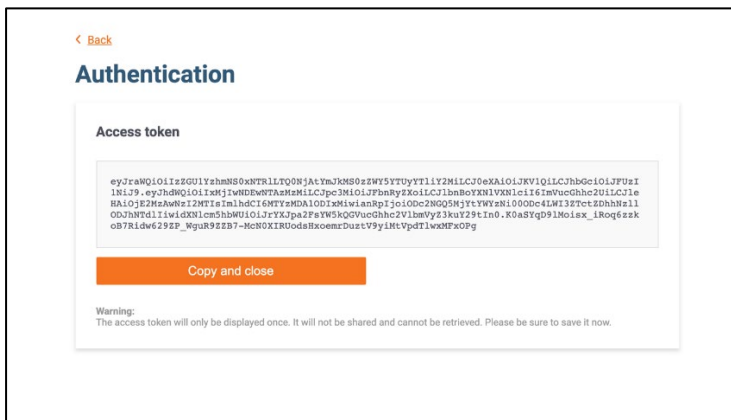
If your Enphase Cloud login is associated with multiple systems or if the system has multiple IQ Gateways, then select the system and the serial number of the IQ Gateway which requires access via local API in the “Select System” and “Select Gateway” drop-downs respectively.

The system name and IQ Gateway serial number can be obtained from the Enphase App as shown in the following images:



The page can be accessed in Enphase App under **Systems->Devices->Gateway** after tapping the menu icon in bottom right corner. The serial number is specified with the label “SN:” under the corresponding IQ Gateway.

- To generate a token, click “Create access token”.



- Copy and paste the token in the home automation setup where access to the IQ Gateway local APIs is required or into the browser where access to the IQ Gateway local UI is required. Save the token securely for future use. Refer to section [Accessing the IQ Gateway using a token](#) for details on how to access the IQ Gateway local UI or local APIs using a token.

NOTE: Tokens are valid for a finite time.

- If the user is a system owner, the token is valid for 1 year.
- If the user is an installer, the token is valid for 12 hours.

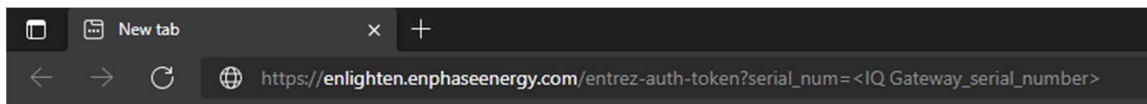
If the credentials used are of installer as well as system owner credentials, i.e., a self-installer, then the web UI based token retrieval outlined in this section will result in a token that is valid for 12 hours. The owner can contact Enphase customer support to change the credentials to the system owner credentials from the installer credentials. Alternately, the owner can use the programmatical route for retrieving tokens outlined in the following section.

Obtaining a token programmatically via GET on a URL

For users who want to retrieve a token programmatically, Enphase provides a URL to retrieve a token and mentions the duration for which the token will be valid.

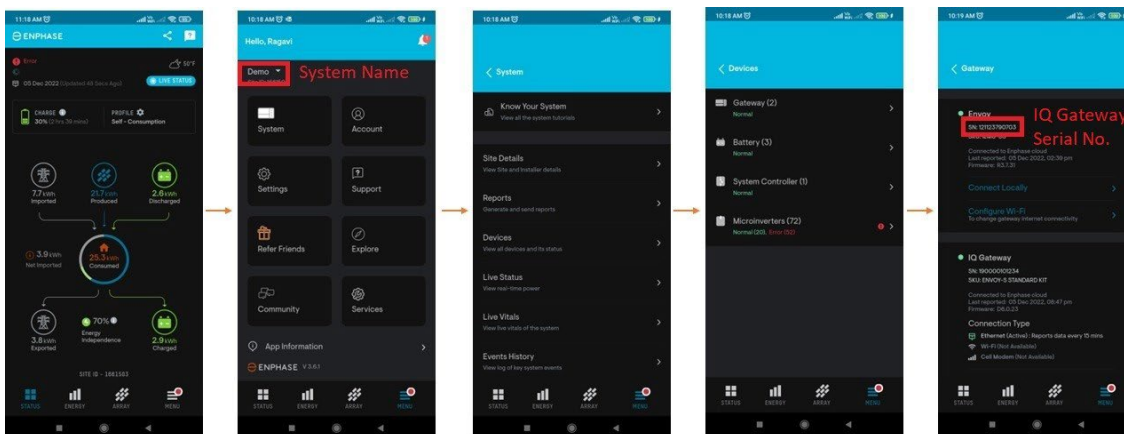
To construct the right URL for the IQ Gateway:

1. Log in to Enphase Cloud (<https://enlighten.enphaseenergy.com>) with the system owner Enphase credentials.
2. Paste the token retrieval URL into the web browser's address bar:
https://enlighten.enphaseenergy.com/entrez-auth-token?serial_num=<IQ Gateway serial number>



NOTE: Replace <IQ Gateway_serial_number> in the above URL with the serial number of the specific IQ Gateway.

The IQ Gateway serial number can be obtained from the Enphase App as shown in the following images:



The page can be accessed in Enphase App under **Systems->Devices->Gateway** after tapping the menu icon in bottom right corner. The serial number is specified with the label “SN:” under the corresponding IQ Gateway.

When the token retrieval URL is accessed via the web browser, the browser does an HTTP GET on the URL. The content of the response can be seen in the browser. This content contains the token as well as the expected expiry date and time in UNIX epoch timestamp format. The duration for which the token is valid can be verified using this information.

The following examples show how to programmatically retrieve the token via the URL, using a Shell script or Python script:

Shell script-based token retrieval

Shell script snippet to programmatically retrieve the token:

```
user='<UserName>'
password='<Password>'
```

```

envoy_serial='<Envoy_Serial_No>'

session_id=$(curl -X POST https://enlighten.enphaseenergy.com/login/login.json? -F
"user[email]=$user" -F "user[password]=$password" | jq -r ".session_id")

web_token=$(curl -X POST https://entrez.enphaseenergy.com/tokens -H "Content-Type:
application/json" -d "{\"session_id\": \"$session_id\", \"serial_num\": \"$envoy_serial\",
\"username\": \"$user\"}")

```

NOTE: Replace the following items in the above Shell script:

- <Envoy_Serial_No> with the serial number of the specific IQ Gateway.
- <UserName> and <Password> with the system owner's credentials.

The variable `web_token` obtained in the last step of the script contains the access token.

Python script-based token retrieval

Python script snippet to programmatically retrieve the token:

```

import json
import requests

user='<UserName>'
password='<Password>'
envoy_serial='< Envoy_Serial_No>'

data = {'user[email]': user, 'user[password]': password}

response = requests.post('https://enlighten.enphaseenergy.com/login/login.json?',
data=data) response_data = json.loads(response.text)

data = {'session_id': response_data['session_id'], 'serial_num': envoy_serial, 'username':
user}

response = requests.post('https://entrez.enphaseenergy.com/tokens', json=data)

token_raw = response.text

```

NOTE: Replace the following items in the above Python script:

- <Envoy_Serial_No> with the serial number of the specific IQ Gateway.
- <UserName> and <Password> with the system owner's credentials.

The variable `token_raw` obtained in the last step of the script contains the access token.

Accessing the IQ Gateway using a token

Once a token is obtained, the IQ Gateway local UI or the local APIs can be easily accessed using this token.

Access the IQ Gateway local UI

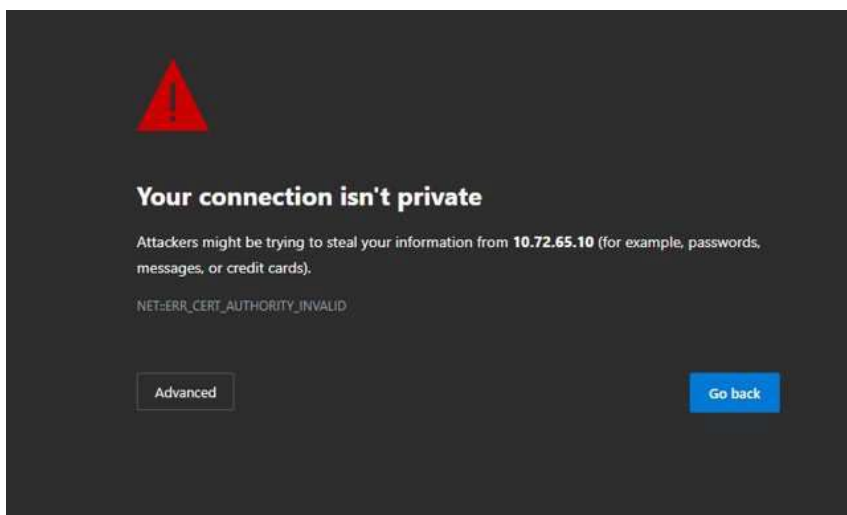
The IQ Gateway local UI can be accessed by the following steps:

1. Open an internet browser (Internet Explorer, Mozilla Firefox, Google Chrome, or Safari) on a computer or mobile device connected to the same Local Area Network (LAN) as the IQ Gateway.
2. If the Gateway has an LCD display, STOP here – this is a legacy Gateway and does not require token-based authentication to access local UI or local APIs.

If the Gateway does not have an LCD display, enter <http://envoy.local/> into the browser.

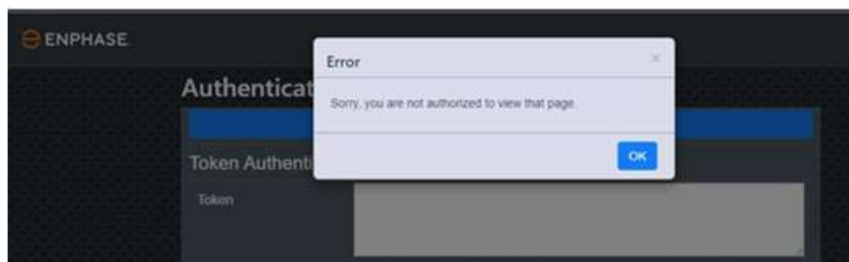
If there are additional Gateway units on the network, you can access them by entering the following strings in the browser window:

- <http://envoy-2.local>
 - <http://envoy-3.local>, and so on.
3. IQ Gateway uses a self-signed certificate. Click “Advanced” → “Continue to <ip_addr of IQ Gateway>(unsafe)”.

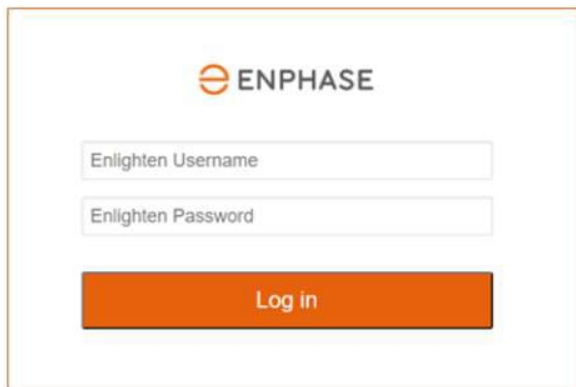


The browser will be redirected to the IQ Gateway authentication page.

4. Click “OK” to proceed if there is any error window popup.

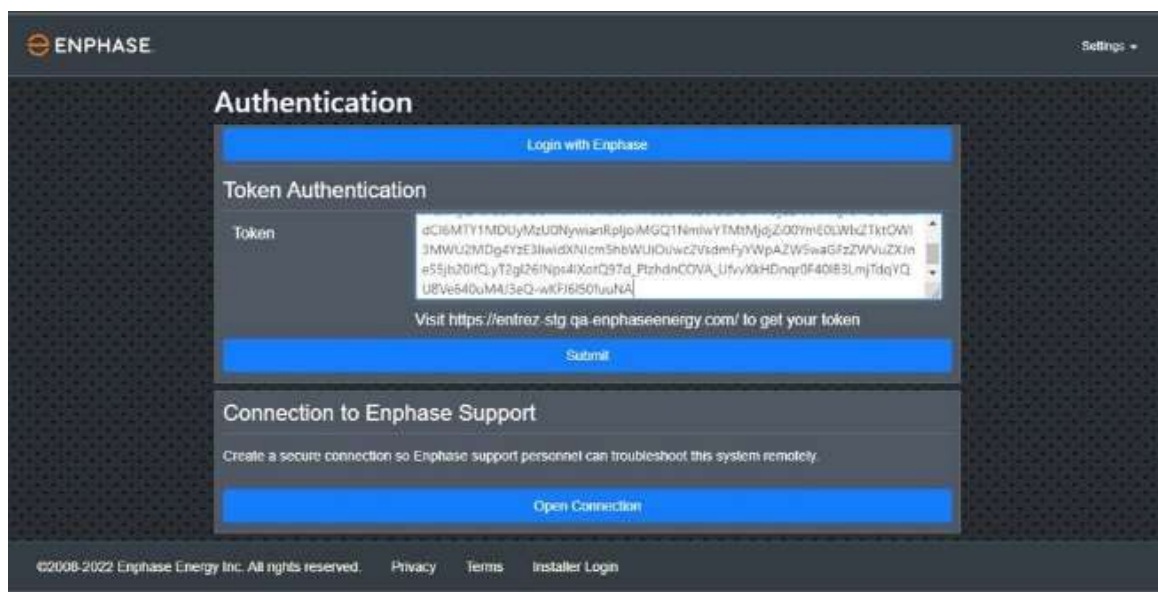


If you are online, click “Login with Enphase” in the authentication screen, enter the system owner’s Enphase Cloud credentials. Authentication with Enphase Cloud happens automatically, and the browser will be redirected to the IQ Gateway page. The token is not required in this case.



If the computer being used to access the IQ Gateway is offline, proceed with step 5.

5. Paste the generated token and click “Submit” to get into the IQ Gateway page. Once the browser has successfully connected with the IQ Gateway, the home screen is displayed in the browser window.



Access the IQ Gateway APIs

The IQ Gateway APIs can be accessed via curl commands with the Authorization Bearer option and HTTPS commands. The following steps describe the way to access IQ Gateway APIs:

1. Connect the computer trying to access the IQ Gateway’s local APIs to the same LAN as the IQ Gateway.
2. Open the Command Prompt on the computer. Check that the router or network host is connected to the IQ Gateway by executing the ping command to IQ Gateway. If ping is successful, proceed with Step 3. If not, check the network connectivity.
3. Enter the following curl command in the Command Prompt to access the IQ Gateway’s available APIs. The format of the curl command is as follows:

curl -f -k -H 'Accept: application/json' -H 'Authorization: Bearer <token code>' -X <API command>

where:

- k: allow self-signed certificate
- H: add http header with token
- X: used to pass method type for API command
- f: fail on error (gives better output in case of unauthenticated redirection)
- L: follow redirects (can use http and allow https redirection)

E.g.: curl -f -k -H 'Accept: application/json' -H 'Authorization: Bearer eyJraWQiOi...' -X GET https://{IQ Gateway-ip}/api/v1/production/inverters

The token is valid only for one year to ensure safety and you must generate a new token upon expiry. The following table lists a few local REST APIs of IQ Gateway.

API Name	Command	Description
Meter details	GET https://{IQ Gateway_ip}/ivp/meters	Returns meter status, type of meter, and number of phase measurements.
Meter readings	GET https://{IQ Gateway_ip}/ivp/meters/readings	Returns measurements from production CT, storage CT and consumption CT, and all are subjected to the availability of CTs.
Inverter production data	GET https://{IQ Gateway_ip}/api/v1/production/inverters	Returns maximum and last reported active power production information of the available microinverters.
Meter's live data	GET https://{IQ Gateway_ip}/ivp/livedata/status	Returns meter's live data with tasks and counters.
Load consumption data	GET https://{IQ Gateway_ip}/ivp/meters/reports/consumption	Returns power consumption information of the loads.

Generate a new token using the methods outlined in section [Generating the token](#) upon expiry. The [Appendix](#) section provides more description about these APIs and a sample JSON response that is obtained from the IQ Gateway when a HTTP GET is done on any of these APIs.

Appendix

Getting meter details

GET https://{ IQ_Gateway_ip}/ivp/meters

Description: Returns meter status, type of meter, and number of phase measurements.

Sample response:


```
[
{
  "eid": 704643328,
  "state": "enabled",
  "measurementType": "production", "phaseMode": "split",
  "phaseCount": 2,
  "meteringStatus": "normal",
  "statusFlags": [ ]
},
{
  "eid": 704643584,
  "state": "enabled",
  "measurementType": "net-consumption", "phaseMode": "split",
  "phaseCount": 2,
  "meteringStatus": "normal",
  "statusFlags": [ ]
}
]
```

Getting meter readings

GET https://{IQ_Gateway_ip}/ivp/meters/readings

Description: Returns measurements from production CT, storage CT and consumption CT, and all are subjected to the availability of CTs. This data will get updated once every 5 minutes.

Sample response:

```
[
{
  "eid": 704643328,
  "timestamp": 1654218661,
  "actEnergyDivd": 1608426.912, "actEnergyRcvd": 4.923,
  "apparentEnergy": 1648123.109, "reactEnergyLagg": 52600.292, "reactEnergyLead":
  19013.342, "instantaneousDemand": 132.118, "activePower": 132.118,
  "apparentPower": 5328.778, "reactivePower": -5328.778,
  "pwrFactor": 0.025,
```

```

"voltage": 246.377,
"current": 43.257,
"freq": 59.188,
"channels": [{
"eid": 1778385169,
"timestamp": 1654218661,
"actEnergyDivd": 803639.138,
"actEnergyRcvd": 2.650,
"apparentEnergy": 823442.481,
"reactEnergyLagg": 26264.291,
"reactEnergyLead": 9545.452,
"instantaneousDemand": 66.037,
"activePower": 66.037,
"apparentPower": 2663.476,
"reactivePower": -2663.476,
"pwrFactor": 0.025,
"voltage": 123.184,
"current": 21.622,
"freq": 59.188
},
{
"eid": 1778385170,
"timestamp": 1654218661,
"actEnergyDivd": 804787.774,
"actEnergyRcvd": 2.273,
"apparentEnergy": 824680.628,
"reactEnergyLagg": 26336.001,
"reactEnergyLead": 9467.890,
"instantaneousDemand": 66.082,
"activePower": 66.082,
"apparentPower": 2665.302,
"reactivePower": -2665.302,
"pwrFactor": 0.025,
"voltage": 123.193,
"current": 21.635,
"freq": 59.188
},
{
"eid": 1778385171,
"timestamp": 1654218661,

```

```

"actEnergyDivd": 0.000,
"actEnergyRcvd": 0.000,
"apparentEnergy": 0.000,
"reactEnergyLagg": 0.000,
"reactEnergyLead": 0.000,
"instantaneousDemand": 0.000,
"activePower": 0.000,
"apparentPower": 0.000,
"reactivePower": 0.000,
"pwrFactor": 0.000,
"voltage": 0.000,
"current": 0.000,
"freq": 59.188
}
]
},
{
"eid": 704643584,
"timestamp": 1654218661,
"actEnergyDivd": 48540.732,
"actEnergyRcvd": 1244797.861,
"apparentEnergy": 1332629.594,
"reactEnergyLagg": 13955.857,
"reactEnergyLead": 30823.381,
"instantaneousDemand": -0.000,
"activePower": -0.000,
"apparentPower": 34.831,
"reactivePower": -0.000,
"pwrFactor": 0.000,
"voltage": 246.338,
"current": 0.283,
"freq": 59.188,
"channels": [{
"eid": 1778385425,
"timestamp": 1654218661,
"actEnergyDivd": 24176.961,
"actEnergyRcvd": 600344.235,
"apparentEnergy": 644044.993,
"reactEnergyLagg": 5391.081,
"reactEnergyLead": 15459.001,

```

```

"instantaneousDemand": -0.000,
"activePower": -0.000,
"apparentPower": 16.858,
"reactivePower": -0.000,
"pwrFactor": 0.000,
"voltage": 123.152,
"current": 0.137,
"freq": 59.188
},
{
"eid": 1778385426,
"timestamp": 1654218661,
"actEnergyDivd": 24363.771,
"actEnergyRcvd": 644453.626,
"apparentEnergy": 688584.601,
"reactEnergyLagg": 8564.776,
"reactEnergyLead": 15364.380,
"instantaneousDemand": -0.000,
"activePower": -0.000,
"apparentPower": 17.973,
"reactivePower": -0.000,
"pwrFactor": 0.000,
"voltage": 123.186,
"current": 0.146,
"freq": 59.188
},
{
"eid": 1778385427,
"timestamp": 1654218661,
"actEnergyDivd": 129399.711,
"actEnergyRcvd": 93791.210,
"apparentEnergy": 242548.385,
"reactEnergyLagg": 15196.459,
"reactEnergyLead": 10272.271,
"instantaneousDemand": 0.000,
"activePower": 0.000,
"apparentPower": 2697.761,
"reactivePower": 2697.761,
"pwrFactor": 0.000,
"voltage": 123.175,

```

```

"current": 21.902,
"freq": 59.188
}
],
{
"eid": 704643840,
"timestamp": 1654218661,
"actEnergyDivd": 258799.422,
"actEnergyRcvd": 187582.421,
"apparentEnergy": 485096.770,
"reactEnergyLagg": 30392.918,
"reactEnergyLead": 20544.543,
"instantaneousDemand": 0.000,
"activePower": 0.000,
"apparentPower": 5395.521,
"reactivePower": 5395.521,
"pwrFactor": 0.000,
"voltage": 246.351,
"current": 43.804,
"freq": 59.188,
"channels": [{
"eid": 1778385681,
"timestamp": 1654218661,
"actEnergyDivd": 129399.711,
"actEnergyRcvd": 93791.210,
"apparentEnergy": 242548.385,
"reactEnergyLagg": 15196.459,
"reactEnergyLead": 10272.271,
"instantaneousDemand": 0.000,
"activePower": 0.000,
"apparentPower": 2697.761,
"reactivePower": 2697.761,
"pwrFactor": 0.000,
"voltage": 123.175,
"current": 21.902,
"freq": 59.188
},
{
"eid": 1778385682,

```

```

"timestamp": 1654218661,
"actEnergyDivd": 129399.711,
"actEnergyRcvd": 93791.210,
"apparentEnergy": 242548.385,
"reactEnergyLagg": 15196.459,
"reactEnergyLead": 10272.271,
"instantaneousDemand": 0.000,
"activePower": 0.000,
"apparentPower": 2697.761,
"reactivePower": 2697.761,
"pwrFactor": 0.000,
"voltage": 123.175,
"current": 21.902,
"freq": 59.188
},
{
"eid": 1778385683,
"timestamp": 1654218661,
"actEnergyDivd": 0.000,
"actEnergyRcvd": 0.000,
"apparentEnergy": 0.000,
"reactEnergyLagg": 0.000,
"reactEnergyLead": 0.000,
"instantaneousDemand": 0.000,
"activePower": 0.000,
"apparentPower": 0.000,
"reactivePower": 0.000,
"pwrFactor": 0.000,
"voltage": 0.000,
"current": 0.000,
"freq": 59.188
}
]
}
]

```

Getting reported inverter production data

GET https://{IQ_Gateway_ip}/api/v1/production/inverters

Description: Returns maximum and last reported active power production information of the available microinverters. This data will get updated once every 5 minutes.

Sample response:

```
[
  {
    "serialNumber": "121935144671",
    "lastReportDate": 1654171836,
    "devType": 1,
    "lastReportWatts": 15,
    "maxReportWatts": 38
  },
  {
    "serialNumber": "121935144623",
    "lastReportDate": 1654171766,
    "devType": 1,
    "lastReportWatts": 5,
    "maxReportWatts": 5
  }
]
```

Getting meter's live data**GET** https://{IQ_Gateway_ip}/ivp/livedata/status**Description:** Returns meter's live data with tasks and counters.**Sample response:**

```
{
  "connection": {
    "mqtt_state": "connected",
    "prov_state": "configured",
    "auth_state": "ok",
    "sc_stream": "enabled",
    "sc_debug": "enabled"
  },
  "meters": {
    "last_update": 1654221647,
    "soc": 100,
    "main_relay_state": 0,
    "gen_relay_state": 5,
    "backup_bat_mode": 1,
    "backup_soc": 10,
    "is_split_phase": 1,

```

```
"phase_count": 0,  
"enc_agg_soc": 100,  
"enc_agg_energy": 24800,  
"acb_agg_soc": 0,  
"acb_agg_energy": 0,  
"pv": {  
  "agg_p_mw": 329549,  
  "agg_s_mva": 329549,  
  "agg_p_ph_a_mw": 329549,  
  "agg_p_ph_b_mw": 0,  
  "agg_p_ph_c_mw": 0,  
  "agg_s_ph_a_mva": 329549,  
  "agg_s_ph_b_mva": 0,  
  "agg_s_ph_c_mva": 0  
},  
"storage": {  
  "agg_p_mw": -220800,  
  "agg_s_mva": -559446,  
  "agg_p_ph_a_mw": -220800,  
  "agg_p_ph_b_mw": 0,  
  "agg_p_ph_c_mw": 0,  
  "agg_s_ph_a_mva": -559446,  
  "agg_s_ph_b_mva": 0,  
  "agg_s_ph_c_mva": 0  
},  
"grid": {  
  "agg_p_mw": 0,  
  "agg_s_mva": 0,  
  "agg_p_ph_a_mw": 0,  
  "agg_p_ph_b_mw": 0,  
  "agg_p_ph_c_mw": 0,  
  "agg_s_ph_a_mva": 0,  
  "agg_s_ph_b_mva": 0,  
  "agg_s_ph_c_mva": 0  
},  
"load": {  
  "agg_p_mw": 108749,  
  "agg_s_mva": -229897,  
  "agg_p_ph_a_mw": 108749,  
  "agg_p_ph_b_mw": 0,
```



```
"agg_p_ph_c_mw": 0,
"agg_s_ph_a_mva": -229897,
"agg_s_ph_b_mva": 0,
"agg_s_ph_c_mva": 0
},
"generator": {
"agg_p_mw": 0,
"agg_s_mva": 0,
"agg_p_ph_a_mw": 0,
"agg_p_ph_b_mw": 0,
"agg_p_ph_c_mw": 0,
"agg_s_ph_a_mva": 0,
"agg_s_ph_b_mva": 0,
"agg_s_ph_c_mva": 0
}
},
"tasks": {
"task_id": 27672012,
"timestamp": 1654219883
},
"counters": {
"main_CfgLoad": 1,
"main_CfgChanged": 1,
"main_taskUpdate": 62,
"MqttClient_publish": 10260,
"MqttClient_live_debug": 190,
"MqttClient_respond": 260,
"MqttClient_msgarrvd": 130,
"MqttClient_create": 13,
"MqttClient_setCallbacks": 13,
"MqttClient_connect": 13,
"MqttClient_connect_err": 5,
"MqttClient_connect_Err": 5,
"MqttClient_subscribe": 8,
"SSL_Keys_Create": 13,
"sc_hdlDataPub": 9440,
"sc_SendStreamCtrl": 72,
"sc_SendDemandRspCtrl": 65517,
"rest_Meters": 7,
"rest_Status": 579
```

```
}
}
```

Getting power consumption data

GET https://{IQ_Gateway_ip}/ivp/meters/reports/consumption

Description: Returns power consumption information of the loads. This data will get updated once every 5 minutes.

Sample response:

```
{
  "createdAt": 1654625079,
  "reportType": "net-consumption",
  "cumulative": {
    "currW": 119.423,
    "actPower": 119.423,
    "apprmtPwr": 105.678,
    "reactPwr": -261.046,
    "whDlvdCum": 43110.122,
    "whRcvdCum": 0.000,
    "varhLagCum": -25071.856,
    "varhLeadCum": 35895.778,
    "vahCum": 192725.807,
    "rmsVoltage": 241.427,
    "rmsCurrent": 0.875,
    "pwrFactor": 1.00,
    "freqHz": 60.00
  },
  "lines": [
    {
      "currW": 56.672,
      "actPower": 56.672,
      "apprmtPwr": 49.248,
      "reactPwr": -136.579,
      "whDlvdCum": 21051.342,
      "whRcvdCum": 0.000,
      "varhLagCum": -12541.347,
      "varhLeadCum": 18473.849,
      "vahCum": 96511.746,
      "rmsVoltage": 120.673,
      "rmsCurrent": 0.408,
      "pwrFactor": 1.00,
    }
  ]
}
```

```

"freqHz": 60.00
},
{
"currW": 62.751,
"actPower": 62.751,
"appmntPwr": 56.430,
"reactPwr": -124.467,
"whDlvdCum": 22058.779,
"whRcvdCum": 0.000,
"varhLagCum": -12530.509,
"varhLeadCum": 17421.929,
"vahCum": 96214.061,
"rmsVoltage": 120.753,
"rmsCurrent": 0.467,
"pwrFactor": 1.00,
"freqHz": 60.00
}
],
{
"createdAt": 1654625079,
"reportType": "net-consumption",
"cumulative": {
"currW": -1905.274,
"actPower": -1905.274,
"appmntPwr": -1920.786,
"reactPwr": -260.398,
"whDlvdCum": -152327.377,
"whRcvdCum": 0.000,
"varhLagCum": 32.752,
"varhLeadCum": 35951.521,
"vahCum": 192725.807,
"rmsVoltage": 241.427,
"rmsCurrent": -15.912,
"pwrFactor": -1.00,
"freqHz": 60.00
},
"lines": [
{
"currW": -954.876,

```

```

"actPower": -954.876,
"appntPwr": -963.431,
"reactPwr": -136.579,
"whDlvdCum": -76608.517,
"whRcvdCum": 0.000,
"varhLagCum": 16.155,
"varhLeadCum": 18488.097,
"vahCum": 96511.746,
"rmsVoltage": 120.673,
"rmsCurrent": -7.984,
"pwrFactor": -1.00,
"freqHz": 60.00
},
{
"currW": -950.398,
"actPower": -950.398,
"appntPwr": -957.355,
"reactPwr": -123.819,
"whDlvdCum": -75718.860,
"whRcvdCum": 0.000,
"varhLagCum": 16.598,
"varhLeadCum": 17463.423,
"vahCum": 96214.061,
"rmsVoltage": 120.753,
"rmsCurrent": -7.928,
"pwrFactor": -1.00,
"freqHz": 60.00
}
]
}

```