Envoy REST Interface

The Envoy has a set of REST interfaces to communicate with external world and vice versa. These interfaces are simple curl commands. The section below provides interfaces classified based categories of modules they act upon.

Envoy Status

curl -X GET http://localhost/ivp/ensemble/submod	To get list of all submod serial number.
curl -X GET http://localhost/ivp/ensemble/dataraw/<Serial number of EnChg>	This will provide telemetry status of all units of encharge and submodules.
cat /etc/emu_versions.txt	current all versions
curl -X GET http://localhost/ivp/ensemble/inventory	To what all devices are connected
curl -X GET http://localhost/ivp/ensemble/status	
curl -X GET http://localhost/ivp/zb/status	ZB Status
curl -X GET http://localhost/ivp/peb/rpttemp	
curl -X GET http://localhost/info	
curl -X GET http://localhost/ivp/ensemble/relay	
curl -X POST http://localhost/ivp/ensemble/relay -d '{"mains_admin_state": "X"}'	Interface to set Grid state Open/Closed X=”open” or ”closed”
curl -X GET http://localhost/ivp/firmware_manager/state	firmware upgrade status

ZB Provision and Status

API	Response type	
BLE: curl -X POST http://localhost/ivp/zb/provision -d '{"serial_nums":[{"serial_num":"serial number of ENC/ENP"}]}' CLI: curl -X POST http://localhost/ivp/zb/provision -d '{"serial_nums":[{"serial_num":"serial number of ENC/ENP"}]}'	400: Any one of the serial number provision failed. 200 OK: All serial provision failed	Register a device serial number and key exchange start
`curl -X DELETE http://localhost/ivp/zb/provision?serial_number=serial_number1,serial_number2`	400: If any one of the serial number delete operation failed. 200 OK: All serial number delete failed.	To delete a device `curl -X DELETE http://localhost/ivp/zb/provision?serial_number=serial_number` To delete multiple devices (Rel 6.0.92) `curl -X DELETE http://localhost/ivp/zb/provision?serial_number=serial_number1`
/var/opt/agf/zigbee_agent_cfg.json		To check the private key and should match Encharge
`curl -X POST http://localhost/ivp/zb/commission`		To seal the commission process.
`curl -X GET http://localhost/ivp/zb/commission`		Get `"/var/opt/agf/zigbee_agent_cfg.json"` file.
`curl -X DELETE http://localhost/ivp/zb/commission`		delete this JSON file.
curl -X GET http://localhost/ivp/zb/pairing_status or psts		Get paring status
curl -X POST http://localhost/ivp/zb/reset_device -d '{"serial_num" : "123456789012"}'		Reset device by serial number

<table>
<tr>
<td><pre>
curl -X POST http://localhost/ivp/zb/provision -d '{
  "serial_nums": [
    { "serial_num":"122038023614",
      "link_key":"0102030405060708090A0B0C0D0E0F00",
      "mac_addr": "0013a200112233",
      "device_type": 13,
      "phase":1 },
    { "serial_num":"122038023605",
      "link_key":"0102030405060708090A0B0C0D0E0F00",
      "mac_addr": "0013a200112233",
      "device_type": 13,
      "phase":2 },
    { "serial_num":"122038023601",
      "link_key":"0102030405060708090A0B0C0D0E0F00",
      "mac_addr": "0013a200112233",
      "device_type": 13,
      "phase":3 },
    { "serial_num":"121938050739",
      "link_key":"0A0B0C0D0E0F00010203040506070809",
      "mac_addr": "0013a209656233",
      "device_type": 17 }
  ]
}'
</pre></td>
<td></td>
<td>provision over ap mode providing phase information for emea</td>
</tr>
<tr>
<td><pre>
curl -X POST http://localhost/ivp/zb/provision -d '{
  "serial_nums": [
    { "serial_num":"122038023614",
      "link_key":"0102030405060708090A0B0C0D0E0F00",
      "mac_addr": "0013a200112233",
      "device_type": 13,
      "phase":1,
      "der_index":1 },
    { "serial_num":"122038023605",
      "link_key":"0102030405060708090A0B0C0D0E0F00",
      "mac_addr": "0013a200112233",
      "device_type": 13,
      "phase":2,
      "der_index":2 },
    { "serial_num":"122038023601",
      "link_key":"0102030405060708090A0B0C0D0E0F00",
      "mac_addr": "0013a200112233",
      "device_type": 13,
      "phase":3,
      "der_index":3 },
    { "serial_num":"121938050739",
      "link_key":"0A0B0C0D0E0F00010203040506070809",
      "mac_addr": "0013a209656233",
      "device_type": 17 }
  ]
</pre></td>
<td></td>
<td>provision api updated for 8.1 convergence release support

for including der index . der index has to be 1 for ens 1.0/1/5 and 2 for ens 2.0</td>
</tr>
</table>

Other supporting commands which can be used for ENC and ENP

pair get_key	To get private stored.
devinfo	to get serial of ENP/ENC
To force ZB communication failure run EnC/EnP CLI	1. zb verbose on 2. znr (reset network) 3. Wait until state changes to WAITING TO JOIN 4. zb at DJ 1 (disable join; so this a few times until all the prints stop)

Grid Profile

curl -X GET http://localhost/ivp/grest/profile/acti	To get the active grid profile.
`curl -X PUT http://localhost/ivp/adc/relaxedparams/add`	Add a JSON to Envoy @ /var/opt/agf/mm_profile.json
`curl -X PUT http://localhost/ivp/adc/relaxedparams/delete`	Delete the /var/opt/agf/mm_profile.json JSON file
http://10.50.60.19/ivp/adc/record?eid=1090519312&agf=xyz	Reecon/agfdump tool and profiles xyz=fpf, vrt, frt
curl -X POST http://localhost/ivp/ensemble/relay -d '{"mains_admin_state": "open/closed"}'	
`curl -X GET http://localhost/ivp/ensemble/relay`	

curl -X GET http://localhost/admin/lib/tariff	Get the details of the active tariff (details like schedule, backup soc etc, PEL etc)

Live Status/Live Vitals

command	
curl -X GET http://localhost/ivp/livedata/status	
`curl -X GET http://localhost/ivp/livedata/counters`	
/ivp/livedata/tasks /ivp/livedata/test `curl -X POSThttp://localhost`/ivp/livedata/stream -d “{ }’	

Meters

<table>
<tr><td>curl -X GET http://localhost/ivp/meters</td><td>GET the status of meters (enabled or disabled , along with configuration :
<pre>
[11:11:42][envoy:~$] curl -X GET http://localhost/ivp/meters
[
    {
        "eid": 704643328,
        "state": "enabled",
        "measurementType": "production",
        "phaseMode": "split",
        "phaseCount": 2,
        "meteringStatus": "normal",
        "statusFlags": []
    },
    {
        "eid": 704643584,
        "state": "enabled",
        "measurementType": "net-consumption",
        "phaseMode": "split",
        "phaseCount": 2,
        "meteringStatus": "normal",
        "statusFlags": []
    }
][11:11:51][envoy:~$]
</pre></td></tr>
<tr><td>curl -X PUT http://localhost/ivp/meters/<meter serial number> -d ‘{“state”: “enabled”}’</td><td>curl -X PUT http://localhost/ivp/meters/704643328 -d ‘{“state”: “enabled”}</td></tr>
<tr><td>curl -X GET http://localhost/ivp/sc/status</td><td></td></tr>
<tr><td>watch curl -s GET http://localhost/ivp/livedata/meters</td><td></td></tr>
</table>

ADC/Inverter related

curl -X GET ‘http://localhost/ivp/adc/record?eid=1090519312	
curl -X GET ‘http://localhost/ivp/adc/record?eid=1090519312&tpm=true’	

To Start/Stop AP MODE

command	**info**	
/etc/init.d/wireless startAP	Start AP	/etc/init.d/wireless {start|stop|startAP|stopAP|test|restart|forget|load|unload} To verify if mode is started then check for file is getting created or not /var/run/wifi/EnvoyAP
/etc/init.d/wireless stopAP	Stop AP	

Site Settings :

command	**description**

/ivp/ss/pcs_settings

get and set the pcs settings in an envoy

parameters :

```
"pcs_settings": {
    "pcsOffering": {
      "PVOversubscription": false,
      "EnchargeOversubscription": false,
      "MPUAvoidance": false
    },
    "mainCircuitBreaker": 0.0,
    "mainPanelBusbar": 0.0,
    "mainPanelDERBreaker": 0.0,
    "pvBreaker": 0.0,
    "enchargeBreaker": 0.0,
    "consumptionMeterLocation": "Between_Main_Load_Panel_and_Enpower",
    "override" : false,
    "max_back_feed" : 0.0
  }
```

/ivp/ss/pel_settings

REST API to read and edit power export limit settings :

parameters :

```
{
  "pel_settings": {
    "PEL": true,
    "Hard_PEL": false,
    "Soft_PEL": true,
    "Export_Limit_Type": "Aggregate",
    "percent": false,
    "apparent": true,
    "PEL_Limit": 3000,
    "Comm_Time_Loss": 120000,
    "Resp_Time_SoftPEL": 0.0,
    "Resp_Time_HardPEL": 0.0,
    "msReset_HardPEL": 0.0,
    "PEL_db": 0.0
  }
}
```

`/ivp/ss/der_settings`

REST API for setting the DER relays :

parameters:

```
"der": {
    "der1": "DER_TYPE_ENCHARGE",
    "der2": "DER_TYPE_NONE"
  },
```

This API is used to define whether a der relay is connected to PV, encharge or nothing.

`/ivp/ss/dry_contact_settings`

REST API for dry contacts

This API is used to provide the settings of dry contacts, the PV serial numbers attached to them, their id, and respective actions.

parameters : The GET API will return an array of four dry contacts with their respective settings :

```
"dry_contacts" : [
185        {
186          "id": "NC1",
187          "type": "NONE/PV/LOAD",
188          "grid_action": "apply/shed/none",
189          "micro_grid_action": "apply/shed/none",
190          "gen_action": "apply/shed/none",
191          "override": "true/false",
192          "load_name":"A/C comp",
193          "mode":"manual/soc",
194          "soc_low":30,
195          "soc_high":70,
196          "pv_serial_nb": ["1234"]
197        },
198        {
199          "id": "NC2",
200          "type": "NONE/PV/LOAD",
201          "grid_action": "apply/shed/none",
202          "micro_grid_action": "apply/shed/none",
203          "gen_action": "apply/shed/none",
204          "override": "true/false",
205          "load_name":"A/C comp",
206          "mode":"manual/soc",
207          "soc_low":30,
208          "soc_high":70,
209          "pv_serial_nb": ["1234"]
210        },
211        {
212          "id": "NO1",
213          "type": "NONE/PV/LOAD",
214          "grid_action": "apply/shed/none",
215          "micro_grid_action": "apply/shed/none",
216          "gen_action": "apply/shed/none",
217          "override": "true/false",
218          "load_name":"A/C comp",
219          "mode":"manual/soc",
220          "soc_low":30,
221          "soc_high":70,
222          "pv_serial_nb": ["1234"]
223        },
224        {
225          "id": "NO2",
226          "type": "NONE/PV/LOAD",
227          "grid_action": "apply/shed/none",
228          "micro_grid_action": "apply/shed/none",
229          "gen_action": "apply/shed/none",
230          "override": "true/false",
231          "load_name":"A/C comp",
232          "mode":"manual/soc",
233          "soc_low":30,
234          "soc_high":70,
235          "pv_serial_nb": ["1234"]
236        }
237      ],


```

The POST API can be used to send settings of each relay one by one.

<table>
<tr>
<td>/ivp/ss/gen_config</td>
<td>

Details and parameters of generator for ensemble 2.0.

GET API response :

```
"generator_settings": {
    "max_cont_gen_amps": "XX",
    "min_gen_loading_perc": "XX",
    "max_gen_efficiency_perc": "XX",
    "name_plate_rating_wat": "XX",
    "start_method": "Auto/Manual",
    "warm_up_mins": "XX",
    "cool_down_mins": "XX",
    "gen_type": "auto/manual/unknown",
    "model": "xxxxxx"
    "manufacturer" : "xxxxxx "
    }

```

POST API :

```
curl -X POST http://localhost/ivp/ss/gen_config -d '{"generator_settings":{"max_cont
```

</td>
</tr>
<tr>
<td>/ivp/ss/gen_mode</td>
<td>

Turn the generator on or off or on auto

```
curl -X POST http://localhost/ivp/ss/gen_mode -d '{"gen_cmd":"off"}'

{

    "gen_cmd": "off/on/auto"

}
```

response :

`"success" or "failure"`

</td>
</tr>
</table>

<table>
<tr>
<td>/ivp/ss/gen_schedule</td>
<td>

GET and POST the settings for generator schedule (and other parameters)

GET Response

```
"exercise_config": {
2       "freq_in_weeks": 4,
3       "start": 540,
4       "duration": 30,
5       "day": "Mon"
6   },
7
8   "default_soc": {
9       "start_soc": 20,
10      "stop_soc": 80
```

</td>
</tr>
</table>

Site Settings :

<table>
<tr>
<th>endpoint</th>
<th>description</th>
</tr>
<tr>
<td>/ivp/ensemble/dry_contacts</td>
<td>

API to get and override the dry contact relay state

operation GET: get the state of all 4 dry contacts

```
{

“dry_contacts“ : [{

“id” : “NO1”,

“status” : “open/closed“,

},

{

“id” : “NO2”,

“status” : “open/closed“,

},

{

“id” : “NC1”,

“status” : “open/closed“,

},

{

“id” : “NC2”,

“status” : “open/closed“,

}]

}
```

operation POST :

Change the state of one of the relays (passed as id) :

```
curl -X POST http://localhost/ivp/ensemble/dry_contacts -d '{"dry_contacts":{"id":"N
```

</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</table>

SCRT :

`/ivp/sc/status`	GET : (no post API) : this API can be used to get the state of the entire system. including : applied profile • active rules • devices serial numbers (PCU's, enpower, encharge) • nameplate ratings • battery schedule • device set points • bias voltages
`/ivp/sc/pvlimit`	Apply a maximum limit on PV produlction
`/ivp/sc/setp_override` (used for debug)	Override the calculated set point
`/ivp/sc/third_party_pv`	This API is added for ensemble-1.7 : settings for third party PV including. : • serial numbers • ac nameplate • does it have free-watt control

EC Agent :

`/ivp/sc/status`	GET : return the list of ensemble devices (encharge and enpower with ll serial numbers, part numbers and grid state) system grid state PV grid state setpoints sent to PV and encharge relay state (open and close for all the relays)

`/ivp/sc/pvlimit`	GET :
	[10:43:42][envoy:~$] curl -X GET http://localhost/ivp/ensemble/devices
	{
	"serial_nums": [
	{
	"serial_num": "121952111098",
	"zigbee_sn": "",
	"zigbee_ic": "",
	"zigbee_crc": "",
	"device_type": 17,
	"commission_time": 1634728035
	},
	{
	"serial_num": "122111031824",
	"zigbee_sn": "",
	"zigbee_ic": "",
	"zigbee_crc": "",
	"device_type": 13,
	"commission_time": 1634728035
	}
	]
	}
	get encharge and enpower device details
	POST : add an encharge to the system

`/ivp/sc/setp_override` (used for debug)	details (serial num, part num, grid state, fw version etc. of encharge and enpower devices
`/ivp/sc/third_party_pv`	Details of subs opponents of the devices (like encharge PCU's)
`/ivp/sc/status`	Details of the freq and volt biases, backup soc, soc etc. from secondary control POST can be used to override the biases for some time (before secondary rewrites them)
`/ivp/sc/pvlimit`	GET or POST the high backup soc limit (100 by default)

<table>
<tr>
<td>

`/ivp/sc/setp_override` (used for debug)

</td>
<td>

GET all the relay status (admin as well as actual) (DER and MID relay)

POST - override the relay to open or close DER or MID :

```
[10:57:38][envoy:~$] curl -X GET http://localhost/ivp/ensemble/relay
{
    "mains_admin_state": "closed",
    "mains_oper_state": "closed",
    "der1_state": 1,
    "der2_state": 0,
    "Enchg_grid_mode": "multimode-ongrid",
    "Solar_grid_mode": "multimode-ongrid"
}
```

</td>
</tr>
<tr>
<td>

`/ivp/sc/third_party_pv`

</td>
<td>

API for the live operational status of the generator

for details see : Interface Control | API for the live operational status of the generator

</td>
</tr>
<tr>
<td>

`/ivp/sc/status`

</td>
<td>

power output of all the encharges. (GET) no POST

eg :

```
}[10:59:58][envoy:~$] curl -X GET http://localhost/ivp/ensemble/power
{
    "devices:": [
        {
            "serial_num": "122111031824",
            "real_power_mw": -307000,
            "apparent_power_mva": -307000,
            "soc": 40
        }
    ]
}[11:03:03][envoy:~$]
```

</td>
</tr>
</table>

ARF

<table>
<tr>
<th></th>
<th></th>
</tr>
<tr>
<td>

/ivp/arf/profile

</td>
<td>

GET the active profile

```
[11:08:17][envoy:~$] curl -X GET http://localhost/ivp/arf/profile
{
    "name": "IEEE 1547 default 2015",
    "id": "5b798662-f374-4633-ac0b-365b1c5b7ea0:0",
    "version": "1.0.7",
    "item_count": 3053
}[11:10:20][envoy:~$]
```

</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</table>

ENS_REST

ivp/ens_rest/inverters_status

GET and PUT

This is created to replace ruby API /installer/agf/inverters_status.

/installer/agf/inverters_status will redirect to this api.

```

[20:38:52][envoy:~$] curl -X GET http://localhost/ivp/ens_rest/inverters_status
[
    {
        "serial_num": "122130026451",
        "device_type": "PCU",
        "device_record_type": "AGF",
        "admin_state": 1,
        "ph_ind": "ph-unk",
        "gcpa_ph_ind": "ph-unk",
        "man_ph_ind": "ph-unk",
        "status": 2,
        "message": "cookie:1,VVAR:1,FRT:1,VRT:1,FPF:1,PRL:1,PLP:1,VW:1,INV2:1,WP:1,
    },
    {
        "serial_num": "122130045711",
        "device_type": "PCU",
        "device_record_type": "AGF",
        "admin_state": 1,
        "ph_ind": "ph-unk",
        "gcpa_ph_ind": "ph-unk",
        "man_ph_ind": "ph-unk",
        "status": 2,
        "message": "cookie:1,VVAR:1,FRT:1,VRT:1,FPF:1,PRL:1,PLP:1,VW:1,INV2:1,WP:1,
    },
    {
        "serial_num": "122130038773",
        "device_type": "PCU"
```