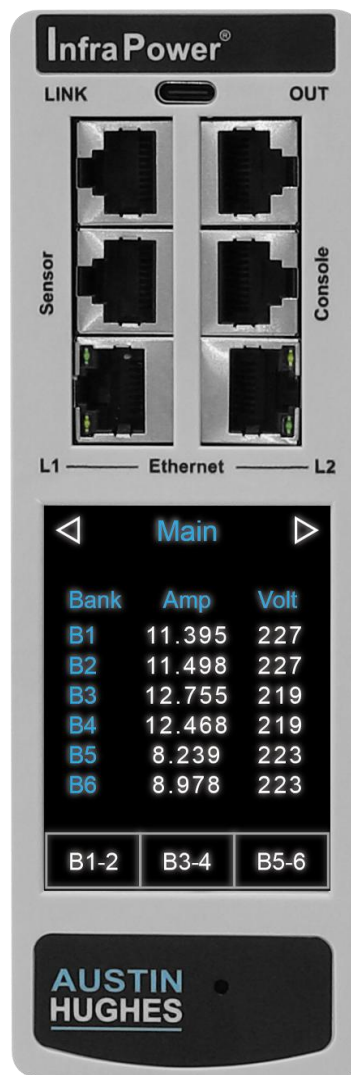


REST API User Guide

Z 2000 Series IP PDU



(Version 1.0)

(1st September, 2025)

Content

< Part 1 > RESTful API

1.1	Introduction	P.1
1.2	GET Operation	P.5
1.3	SET Operation	P.6
1.4	RESET Operation	P.7
1.5	Special(login) Operation	P.8
1.6	Return code	P.9

< Part 2 > Authentication (local-user) P.10

< Part 3 > System P.12

< Part 4 > Network P.16

< Part 5 > Device

5.1	PDU's	P.19
5.2	Circuits	P.22
5.3	Outlets	P.30
5.4	Sensors	P.38

< Part 1 > RESTful API

< 1.1 > Introduction

RESTful API is designed to provide developers and integrators an easy to use method to communicate with the device. All of the device's actions can be accessed through this API. It is all HTTPs POST with JSON as the underlying data structure.

URL : https://{device-ip}/api/v2/<path>

Request : POST

Header :

Authorization : Basic {base64-encoded}

Content-Type : application/json

Body :

```
{
  "token" : <token>,
  "cmd" : "get" | "set" | "login"
  "data" : { ... }
}
```

Response :

```
{
  "ret-code": 0/<return-code>,
  "ret-msg" : null/"<return-message>",
  "data" : { ... }
}
```

Definitions

<token> : A token is assigned to the session after successful authentication.

<login-name> : login user name

<login-pass> : login user password

<return-code> : The result of the request, 0: success, else it's the error code.

<return-msg> : A short description of the error if error is occurred.

<level> : The PDU level, Possible value is 1 to 32.

<circuit-index>: The circuit index, possible value is 1 to 12. (Depends on the PDU model)

<outlet-index>: The outlet index, possible value is 1 to 72. (Depends on the PDU model)

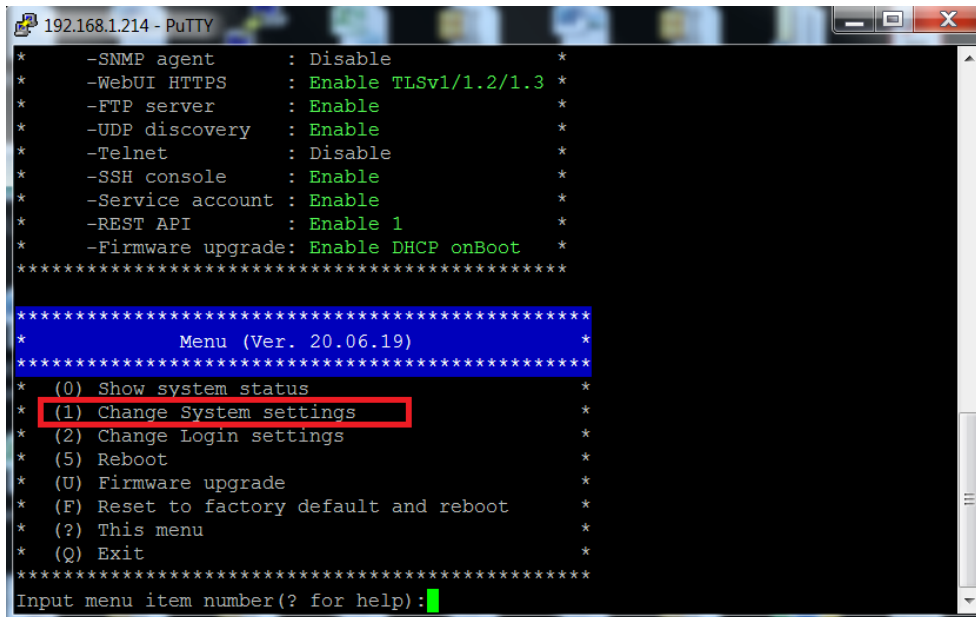
<sensor-index>: The sensor index, possible value is 1 to 16. (Depends on the PDU model)

By default, RESful API is enabled with RW mode which requires authentication for Read / Write operation.

You can change different operation mode or disable RESTful API function using Command Line Interface (CLI) via SSH client such as PuTTY.

To change the RESTful API setting :

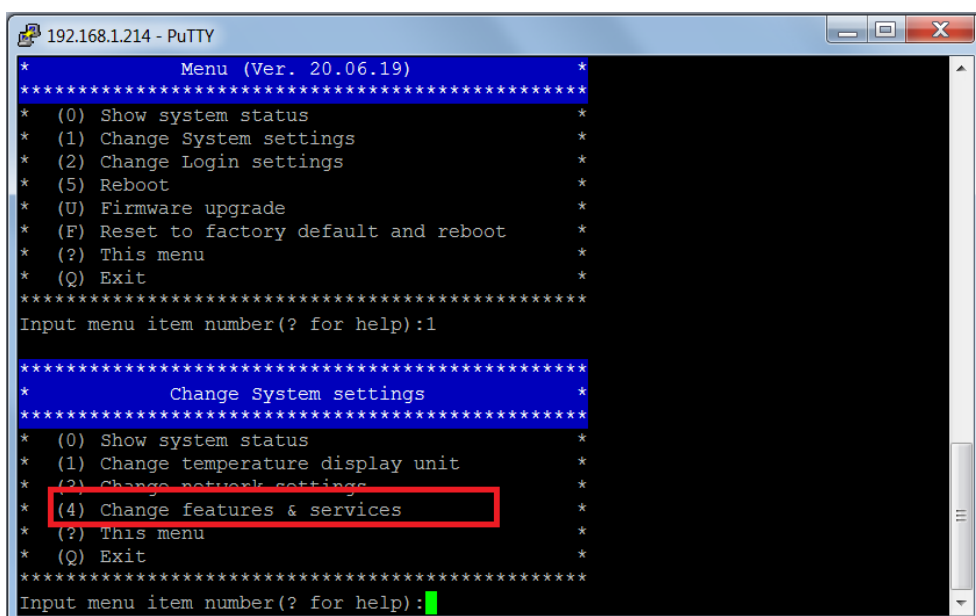
- Step 1. Access the CLI using Putty.
- Step 2. Input the local user's login name and password of PPS-04-S to login the CLI.
- Step 3. Select “ (1) Change System settings “



```

192.168.1.214 - PuTTY
* -SNMP agent      : Disable *
* -WebUI HTTPS     : Enable TLSv1/1.2/1.3 *
* -FTP server      : Enable *
* -UDP discovery   : Enable *
* -Telnet          : Disable *
* -SSH console     : Enable *
* -Service account : Enable *
* -REST API        : Enable 1 *
* -Firmware upgrade: Enable DHCP onBoot *
*****
***** Menu (Ver. 20.06.19) *****
*****
* (0) Show system status *
* (1) Change System settings *
* (2) Change Login settings *
* (5) Reboot *
* (U) Firmware upgrade *
* (F) Reset to factory default and reboot *
* (?) This menu *
* (Q) Exit *
*****
Input menu item number(? for help):
  
```

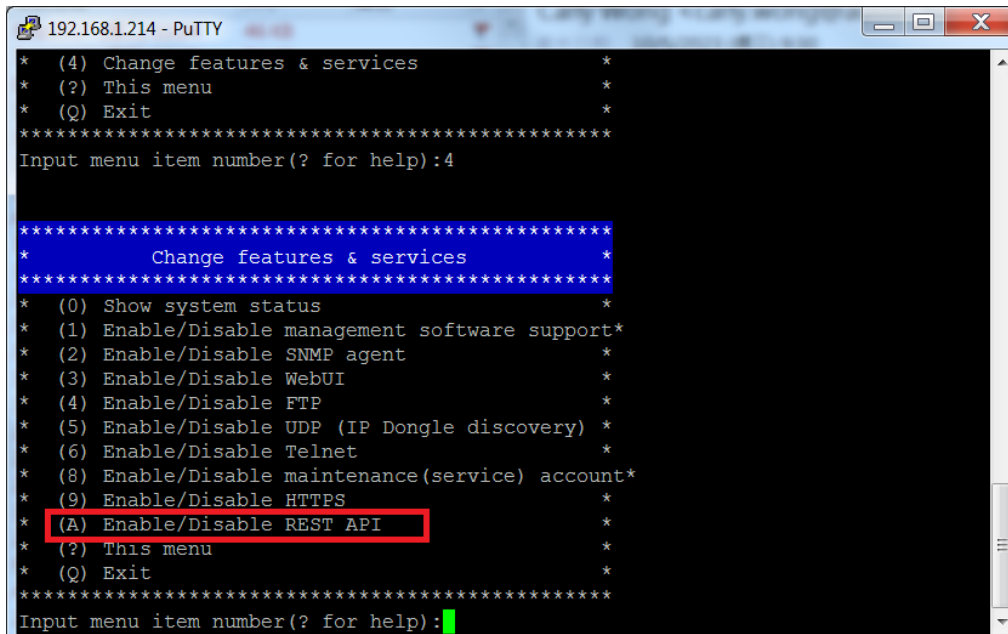
- Step 4. Select “ (4) Change features & services “



```

192.168.1.214 - PuTTY
***** Menu (Ver. 20.06.19) *****
*****
* (0) Show system status *
* (1) Change System settings *
* (2) Change Login settings *
* (5) Reboot *
* (U) Firmware upgrade *
* (F) Reset to factory default and reboot *
* (?) This menu *
* (Q) Exit *
*****
Input menu item number(? for help):1
*****
***** Change System settings *****
*****
* (0) Show system status *
* (1) Change temperature display unit *
* (2) Change network settings *
* (4) Change features & services *
* (?) This menu *
* (Q) Exit *
*****
Input menu item number(? for help):
  
```

Step 5. Select “ (A) Enable/Disable REST API “



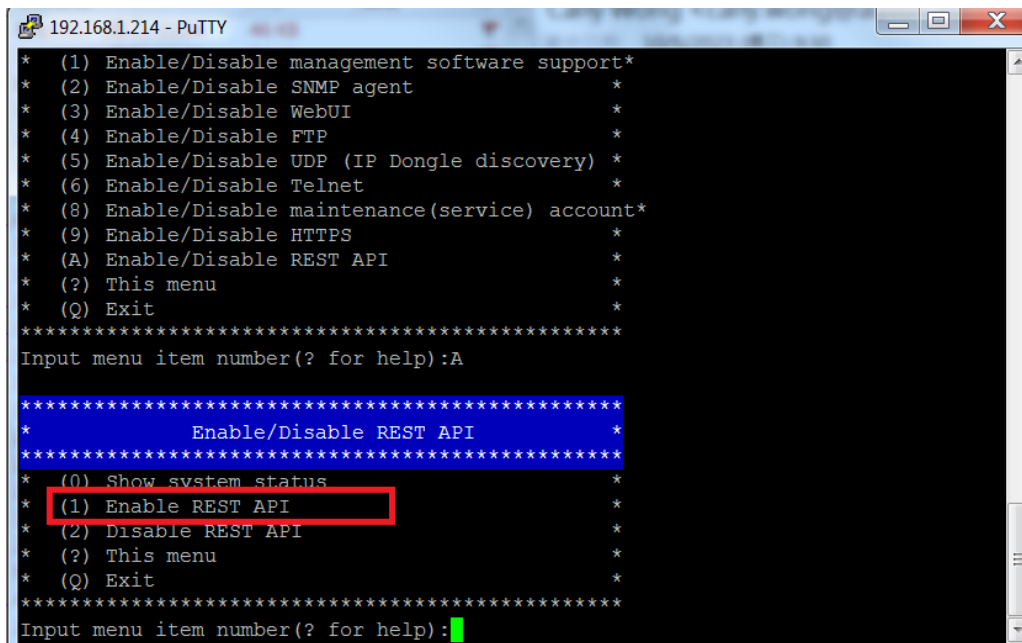
```

192.168.1.214 - PuTTY
* (4) Change features & services *
* (?) This menu *
* (Q) Exit *
*****
Input menu item number(? for help):4

*****
* Change features & services *
*****
* (0) Show system status *
* (1) Enable/Disable management software support*
* (2) Enable/Disable SNMP agent *
* (3) Enable/Disable WebUI *
* (4) Enable/Disable FTP *
* (5) Enable/Disable UDP (IP Dongle discovery) *
* (6) Enable/Disable Telnet *
* (8) Enable/Disable maintenance(service) account*
* (9) Enable/Disable HTTPS *
* (A) Enable/Disable REST API *
* (?) This menu *
* (Q) Exit *
*****
Input menu item number(? for help):
  
```

Step 6. If you want to disable RESTful API, Select “ (2) Disable REST API “.

If you want to change the operation mode of the RESTful API, select “ (1) Enable REST API “.

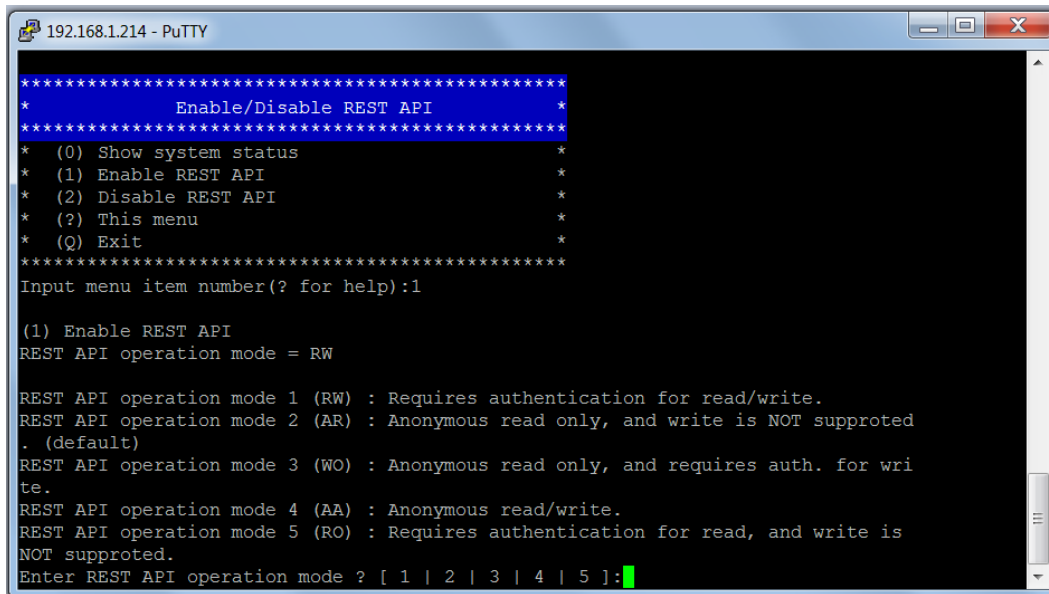


```

192.168.1.214 - PuTTY
* (1) Enable/Disable management software support*
* (2) Enable/Disable SNMP agent *
* (3) Enable/Disable WebUI *
* (4) Enable/Disable FTP *
* (5) Enable/Disable UDP (IP Dongle discovery) *
* (6) Enable/Disable Telnet *
* (8) Enable/Disable maintenance(service) account*
* (9) Enable/Disable HTTPS *
* (A) Enable/Disable REST API *
* (?) This menu *
* (Q) Exit *
*****
Input menu item number(? for help):A

*****
* Enable/Disable REST API *
*****
* (0) Show system status *
* (1) Enable REST API *
* (2) Disable REST API *
* (?) This menu *
* (Q) Exit *
*****
Input menu item number(? for help):
  
```

Step 7. You can select different operation mode based on the requirement.



```

192.168.1.214 - PuTTY
*****
*          Enable/Disable REST API          *
*****
* (0) Show system status                    *
* (1) Enable REST API                      *
* (2) Disable REST API                    *
* (?) This menu                          *
* (Q) Exit                                *
*****
Input menu item number(? for help):1

(1) Enable REST API
REST API operation mode = RW

REST API operation mode 1 (RW) : Requires authentication for read/write.
REST API operation mode 2 (AR) : Anonymous read only, and write is NOT supported
. (default)
REST API operation mode 3 (WO) : Anonymous read only, and requires auth. for write.
REST API operation mode 4 (AA) : Anonymous read/write.
REST API operation mode 5 (RO) : Requires authentication for read, and write is
NOT supported.
Enter REST API operation mode ? [ 1 | 2 | 3 | 4 | 5 ]:

```

< 1.2 > GET Operation

```

Path           : <path>
Body          :
  {
    "token"      : "<token>",
    "cmd"        : "get"
    "data"       : { ... }
  }

```

In order to perform a get, the client sends a post to the desired path on the server. The path can be arbitrary as long as it can be resolved to an object or component in the API structures defined below. The get will return a JSON structure starting at the depth indicated by the path and traversing down to the leaf objects.

Example 1 : get name of pdu level 1

Path : /pdu/1/name

Example GM1) get name of pdu at level 1

PATH: https://192.168.0.1/api/v2/pdus/1/name

BODY:

```

{
  "cmd" : "get",
  "token": "<token>"
}

```

RESPONSE

```

{
  "ret-code": 0,
  "ret-msg": "",
  "data": {
    "pdus": [
      {
        "alive": 1,
        "level": 1,
        "setting": {
          "name": "default_pdu_name"
        }
      }
    ]
  }
}

```

< 1.3 > SET Operation

```

Path           : <path>
Body          :
    {
        "token"   : "<token>",
        "cmd"     : "set",
        "data"    : { ... }
    }

```

In order to perform a set, the client sends a post to the desired path on the server. The path can be arbitrary as long as it can be resolved to a settable object or component in the API structures defined below. The data field of the post contains a JSON structure indicating which objects are to be changed and what the new values are. Any missing objects are left unmodified on the server side. Sets can only be performed at a given depth and sub-objects are not allowed. The response object indicates if there were any errors and no extra data is returned.

Example SM1) set name of pdu at level 1

PATH: https://192.168.0.1/api/v2/pdus/1/name

BODY:

```

{
    "cmd" : "set",
    "token": "<token>",
    "data" : { "name" : "my_pdu_name" }
}

```

RESPONSE

```

{
    "ret-code": 0,
    "ret-msg": ""
}

```

< 1.4 > RESET Operation

```

Path      : <path>
Body      :
{
    "token"   : "<access-token>",
    "cmd"     : "reset",
    "data"    : { ... }
}

```

In order to perform a reset, the client sends a post to the desired path on the server. The path can be arbitrary as long as it can be resolved to a resettable object or component in the API structures defined below. The data field of the post contains a JSON structure indicating which objects are to be changed and what the new values are. Any missing objects are left unmodified on the server side. Resets can only be performed at a given depth and sub-objects are not allowed. The response object indicates if there were any errors and no extra data is returned.

Example RM1) reset circuit 1's peak of pdu at level 16

PATH: <https://192.168.0.1/api/v2/pdus/16/circuits/1/peak>

BODY:

```

{
    "cmd" : "reset",
    "token": "<token>"
}

```

RESPONSE

```

{
    "ret-code": 0,
    "ret-msg": ""
}

```

< 1.5 > Special(Login) Operation

```

Path           : <path>
Header         : Authorization : Basic <base64_encoded>
Body          :
    {
        "cmd"    : "login"
    }

```

Special commands are used to perform other operations that are not covered by sets or gets. Each object may have a set of valid operations along with specific requirements for the data sent. The request and response format are the same as with the sets and gets. Special operations and their parameters are only valid on the path that defines them

< 1.6 > Return code

Return-code :

```
{
  Success:{
    0 – Success
  },
  Authentication-Errors:{
    1001 - Not Authorized
    1002 - Not Authorized: Session expired
    1003 - Not Authorized: Not enough permissions
    1004 - Not Authorized: Invalid password
    1005 - User not found
    1006 – Invalid token
    1007 – Operation is not supported
  }
  Path Errors:{
    3000 - Invalid path
    3001 - Path not found
    3002 - Identifier not found
    3003 - Field not applicable
  }
  Validation Errors:{
    4008 - Invalid url
    4018 – Invalid method
    4019 - Invalid operation
  }
  Device Errors:{
    8000 - Invalid object
    8001 - Invalid object-data
    8002 - Invalid model or model unsupported
    8003 – General error
    8004 – Invalid input
    8005 – operation is done with error(s)
    8006 – Invalid object or unsupported operation
  }
}
```

< Part 2 > Authentication (local-user)

Operation : login | set | get

Object : login-name | login-pass

login-name : the login name(account).

Max. Data Length is 16.

Valid input : 0 ~ 9, A ~ Z, a ~ z, (_), (-), (.)

login-pass: the login password

Max. Data Length is 16

Valid input : 0 ~ 9, A ~ Z, a ~ z & all special characters

except (\), ("), (`) & (')

Example L1) login the get the token

PATH: <https://192.168.0.1/api/v2/auth/local-user>

HEADER:

Content-Type: application/json

Authorization : Basic <base64_encoded>

BODY:

```
{
  "cmd" : "login"
}
```

RESPONSE

```
{
  "ret-code": 0,
  "ret-msg": "",
  "data": {
    "token": "<token>"
  }
}
```

Example L2) logout

PATH: <https://192.168.0.1/api/v2/auth/local-user>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "logout"
}
```

RESPONSE

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

Example L3) update the login name and password

PATH: <https://192.168.0.1/api/v2/auth/local-user>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "set",
  "token": "<token>",
  "data" : { "login-name" : "<login-name>", "login-pass" : "<login-pass>" }
}
```

RESPONSE

```
{
  "ret-code": 0,
  "ret-msg": "",
  "data" : { "token" : "<token>" }
}
```

< Part 3 > System

Operation : reboot | reset | set | get

Object : name | location | temperature-unit | datetime | time-zone | time-setting |
ntp-server | web-protocol | web-port | web-ssl-cert (get only) |
firmware (get only)

name : the name of the ZIP.

Location : the location of the ZIP.

temperature-unit : the display unit for temperature data. Possible data is "C" or "F".

datetime : the calendar and time setting. Data format is "yyyy/mm/dd hh:nn:ss".

time-zone : the time zone setting. (Ref. to Appendix Timezone list)

time-sync : sync. with NTP server. Possible data is "disable" or "enable".

ntp-server : the NTP server to auto sync. System time of the ZIP PDU.

web-protocol : HTTP or HTTPS support. Possible data is "HTTP" or "HTTPS".

web-port : web access port (default 80(HTTP), 443(HTTPS)).

web-ssl-cert : use default or custom SSL cert. Possible data is "default" or "custom".

Firmware : the firmware version.

Example S1) get system information of the ZIP

PATH: <https://192.168.0.1/api/v2/system>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "get",
  "token": "<token>"
}
```

RESPONSE

```
{
  "ret-code": 0,
  "ret-msg": "",
  "data": {
    "system": {
      "name": "default_z4m_name",
      "location": "default_z4m_loc.",
      "temperature-unit": "C",
      "datetime": "2025-08-14 11:05:38",
      "time-zone": "GMT+08:00",
      "time-sync": "enable",
      "ntp-server": "hk.pool.ntp.org",
```

```

    "web-protocol": "HTTPS",
    "web-port": "443",
    "web-ssl-cert": "default",
    "firmware": "Z4M-Z105-250822"
  }
}
}

```

Example S2) get system name of the ZIP

PATH: <https://192.168.0.1/api/v2/system/name>

HEADER:

Content-Type: application/json

BODY:

```

{
  "cmd" : "get",
  "token": "<token>"
}

```

RESPONSE

```

{
  "ret-code": 0,
  "ret-msg": "",
  "data": {
    "system": {
      "name": "default_z4m_name"
    }
  }
}

```

Example S3) update the system.name of the ZIP.

PATH: <https://192.168.0.1/api/v2/system/name>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "set",
  "token": "<token>",
  "data": { "name": "my_ipd_name" }
}
```

RESPONSE

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

Example S4) reboot the ZIP.

PATH: <https://192.168.0.1/api/v2/system>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "reboot",
  "token": "<token>"
}
```

RESPONSE

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

Example S5) reset the ZIP to factory default state.

Warning!!

- Please don't power off the ZIP while processing.
- You may have to change the default password via the WEbUI.
- The default user name and password are 8 zeroes "00000000".
- The default network config is dual lan mode:
 - LAN-1 : IPv4 DHCP
 - LAN-2 : IPv4 DHCP

Remarks : 192.168.0.1 & 192.168.11.1 will be assigned automatically to LAN 1 & LAN 2 respectively if not connected to DHCP server)

- The reset process may take several minutes to be completed.

PATH: <https://192.168.0.1/api/v2/system>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "reset",
  "token": "<token>"
}
```

RESPONSE

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

< Part 4 > Network

Operation : set | get

Object :

```
"lan-1",
"lan-2",
"wireless",
"lan",
"fail-over",
"dns",

"dns-server",
"auto-config",
"dns-1",
"dns-2",

"name",
"mac",
"link",
"speed",
"dhcp",
"ipv4",
"smv4",
"gwv4",
"ipv6",
```

Settable objects : dhcp, ipv4, smv4, gwv4

Example N1) get network information of the ZIP

PATH: <https://192.168.0.1/api/v2/network>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "get",
  "token": "<token>"
}
```

RESPONSE

```
{
  "ret-code": 0,
```

```

"ret-msg": "",
"data": {
  "network": {
    "mode": "dual",
    "fail-over": "0",
    "dns": {
      "auto-config": "disable",
      "dns-server": [
        {
          "name": "Primary DNS",
          "ipv4": "8.8.8.8"
        },
        {
          "name": "Seconday DNS",
          "ipv4": "0.0.0.0"
        }
      ]
    },
  },
  "wired": [
    {
      "name": "lan-1",
      "mac": "20:0a:0d:ff:a6:1c",
      "link": "up",
      "speed": "1000",
      "dhcp": "disable",
      "ipv4": "192.168.11.1",
      "smv4": "255.255.255.0",
      "gwv4": "192.168.11.254",
      "ipv6": "::ffff:c0a8:b1/120"
    },
    {
      "name": "lan-2",
      "mac": "20:0a:0d:ff:99:3f",
      "link": "down",
      "speed": "10",
      "dhcp": "disable",
      "ipv4": "192.168.0.1",
      "smv4": "255.255.255.0",
      "gwv4": "192.168.0.254",
      "ipv6": "::ffff:c0a8:1/120"
    }
  ]
}

```

```
}
]
}
}
}
```

Example N2) set the network.lan-2 of the ZIP.

Warning!!

- Network connection may be disconnected.

PATH: <https://192.168.0.1/api/v2/network/lan-2>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "set",
  "token": "<token>",
  "data": {
    "ipv4": "192.168.0.11",
    "smv4": "255.255.255.0",
    "gwv4": "192.168.0.250"
  }
}
```

RESPONSE

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

< Part 5 > Device

< 5.1 > PDUs

Operation : set | get

<pdu-level> : the cascade level of the pdu, possible data are 1 to 32

Object :

```
"level",
"model",
"uid",
"serial-no",
"name",
"location",
"group-id",
"voltage",
"current",
"power-factor",
"active-power",
"apparent-power",
"cumulative-energy",
"real-time-clock",

"cirucits",
"sensors",
"outlets"
```

Settable objects : name, location

Example P1) get details of the pdu at level 16.

PATH: <https://192.168.0.1/api/v2/pdus/16>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "get",
  "token": "<token>"
}
```

RESPONSE

```
{
  "ret-code": 0,
```

```

"ret-msg": "",
"data": {
  "pdus": [
    {
      "level": 16,
      "alive": 1,
      "setting": {
        "model": "V24C13/6C19-32A-Zi",
        "uid": "2E780016",
        "serial-no": "00020210708-0930-P07F",
        "name": "default_pdu_name",
        "location": "default_pdu_loc.",
        "group-id": "050"
      },
    },
    "status": {
      "voltage": 219.9,
      "current": 0,
      "power-factor": 1,
      "active-power": 0,
      "apparent-power": 0,
      "cumulative-energy": 0,
      "real-time-clock": "1501172010306"
    }
  ]
}

```

Example P2) set name of pdu (name(test_pdu_name 16) of pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "set",
  "token": "<token>",
  "data": {
    "pdus": [
      {
        "level": 16,
        "setting" :{
          "name" : "test_pdu_name 16"
        }
      }
    ]
  }
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

< 5.2 > Circuits

Operation : set | get | reset

<circuit-id> : the circuit id, possible data are 1 to 12

Object :

```
"index",
"rated-amp",
"rated-volt",
"name",
"amp-state",
"voltage",
"current",
"power-factor",
"active-power",
"apparent-power",
"peak",
"peak-timestamp",
"cumulative-energy",
"cum-timestamp",
"protection-state",
"alarm",
"rising-alert",
"low-alert"
```

Settable objects : alarm, rising-alert, low-alert

Resettable objects : peak, cumulative-energy

Example C1) get details of the circuits at pdu level 16.

PATH: <https://192.168.0.1/api/v2/pdus/16/circuits>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "get",
  "token": "<token>"
}
```

RESPONSE

```
{
  "ret-code": 0,
  "ret-msg": "",
  "data": {
    "pdus": [
      {
        "level": 16,
        "alive": 1,
        "circuits": [
          {
            "index": 1,
            "setting": {
              "rated-amp": 16,
              "name": "Circuit A",
              "alarm": 12.8,
              "rising-alert": 0,
              "low-alert": 0
            },
            "status": {
              "amp-state": 0,
              "voltage": 216.1,
              "current": 0,
              "power-factor": 1,
              "active-power": 0,
              "apparent-power": 0,
              "peak": 0,
              "peak-timestamp": "150117172150",
              "cumulative-energy": 0,
              "cum-timestamp": "150101000000",
              "protection-state": 0
            }
          },
          {
            "index": 2,
            "setting": {
              "rated-amp": 16,
              "name": "Circuit B",
              "alarm": 10.0,

```

```

    "rising-alert": 0,
    "low-alert": 0
  },
  "status": {
    "amp-state": 0,
    "voltage": 216.2,
    "current": 0,
    "power-factor": 1,
    "active-power": 0,
    "apparent-power": 0,
    "peak": 0,
    "peak-timestamp": "150117172150",
    "cumulative-energy": 0,
    "cum-timestamp": "150101000000",
    "protection-state": 0
  }
}
]
}
]
}
}

```

Example C2) reset peak of circuit (peak rest of circuit 1 on pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd": "reset",
  "token": "<token>",
  "data": {
    "pdus": [
      {
        "level": 16,
        "circuits": [
          {
            "index": 1,
            "status": {
              "peak": "reset"
            }
          }
        ]
      }
    ]
  }
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

Example C3) reset kwh(cumulative-energy) of circuit (kwh reset of circuit 1 on pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd": "reset",
  "token": "<token>",
  "data": {
    "pdus": [
      {
        "level": 16,
        "circuits": [
          {
            "index": 1,
            "status": {
              "cumulative-energy": "reset"
            }
          }
        ]
      }
    ]
  }
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

Example C4) set alarm threshold of circuit (alarm threshold(10A) of circuit 1 on pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd": "set",
  "token": "<token>",
  "data": {
    "pdus": [
      {
        "level": 16,
        "circuits": [
          {
            "index": 1,
            "setting": {
              "alarm": 10
            }
          }
        ]
      }
    ]
  }
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

Example C5) set rising alert threshold of circuit (rising alert threshold(5A) of circuit 1 on pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd": "set",
  "token": "<token>",
  "data": {
    "pdus": [
      {
        "level": 16,
        "circuits": [
          {
            "index": 1,
            "setting": {
              "rising-alert": 5
            }
          }
        ]
      }
    ]
  }
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

Example C6) set low alert threshold of circuit (low alert threshold(0.001A) of circuit 1 on pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd": "set",
  "token": "<token>",
  "data": {
    "pdus": [
      {
        "level": 16,
        "circuits": [
          {
            "index": 1,
            "setting": {
              "low-alert": 0.001
            }
          }
        ]
      }
    ]
  }
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

< 5.3 > Outlets

Operation : set | get | reset |

<outlet-id> : the outlet id, possible data are 1 to 72.

Object :

```
"index",
"binding-circuit",
"type",
"name",
"rated-volt",
"rated-amp",
"relay-state",
"amp-state",
"voltage",
"current",
"power-factor",
"active-power",
"apparent-power",
"peak",
"cumulative-energy",
"power-up-sequence-delay",
"alarm",
"rising-alert",
"low-alert"
```

Settable objects : name,
 relay-state (On / Off / Power-cycle), power-up-sequence-delay,
 alarm, rising-alert, low-alert

Resettable objects : peak, cumulative-energy

Example O1) get details of the outlets at pdu level 16.

PATH: <https://192.168.0.1/api/v2/pdus/16/outlets>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "get",
  "token": "<token>"
}
```

RESPONSE

```
{
  "ret-code": 0,
  "ret-msg": "",
  "data": {
    "pdus": [
      {
        "level": 16,
        "alive": 1,
        "outlets": [
          {
            "index": 1,
            "setting": {
              "binding-circuit": 1,
              "type": 1,
              "name": "outlet_name_01",
              "alarm": 5,
              "rising-alert": 0,
              "low-alert": 0,
              "power-up-sequence-delay": 1
            },
            "status": {
              "current": 0,
              "relay-state": "On",
              "amp-state": "Normal",
              "voltage": 216.3,
              "peak": 0,
              "power-factor": 1,
              "active-power": 0,
              "apparent-power": 0,
              "cumulative-energy": 0,
            }
          }
        ]
      }
    ]
  }
}
```

Example O2) get specified object (name of outlet 2 on pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus/16/outlets/2/name>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd": "get",
  "token": "<token>"
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": "",
  "data": {
    "pdus": [
      {
        "alive": 1,
        "level": 16,
        "outlets": [
          {
            "index": 2,
            "setting": {
              "name": "test_out_name02"
            }
          }
        ]
      }
    ]
  }
}
```

Example O3) set specified object (name of outlet 2 on pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus/16/outlets/2/name>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "set",
  "token": "<token>",
  "data" : { "name" : "test_out_name02" }
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

Example O4) set specified object using data (name of outlet 2 on pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "set",
  "token": "<token>",
  "data": {
    "pdus": [
      {
        "level": 16,
        "outlets": [
          {
            "index": 2,
            "setting": {
              "name": "set_out_name02"
            }
          }
        ]
      }
    ]
  }
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

Example 05) set multiple objects using data (name of outlet 2, 3 on pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd": "set",
  "token": "<token>",
  "data": {
    "pdus": [
      {
        "level": 16,
        "outlets": [
          {
            "index": 2,
            "setting": {
              "name": "set_out_name02"
            }
          },
          {
            "index": 3,
            "setting": {
              "name": "set_out_name03"
            }
          }
        ]
      }
    ]
  }
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

Example O6) set specified object using data (switch off relay of outlet 2 on pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd": "set",
  "token": "<token>",
  "data": {
    "pdus": [
      {
        "level": 16,
        "outlets": [
          {
            "index": 2,
            "status": {
              "relay-state": "off"
            }
          }
        ]
      }
    ]
  }
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

Example 07) set specified object using data (alarm threshold to 8.5A of outlet 2 on pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd": "set",
  "token": "<token>",
  "data": {
    "pdus": [
      {
        "level": 16,
        "outlets": [
          {
            "index": 2,
            "setting": {
              "alarm": 8.5
            }
          }
        ]
      }
    ]
  }
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

< 5.4 > Sensors

Operation : set | get

<sensor-id> : the sensor id, possible data are 1 to 16.

Object :

```
"index",
"name",
"location",
"type",
"activation",
"connect-state",
"alarm-state",
"temp-state",
"temp-measurement",
"temp-alarm",
"temp-rising-alert",
"temp-low-alert",
"humid-state",
"humid-measurement",
"humid-alarm",
"humid-rising-alert",
"humid-low-alert"
```

Settable objects : location,
activation
temp-alarm, temp-rising-alert,
humid-alarm, humid-rising-alert

Example S1) get details of the sensors at pdu level 16.

PATH: <https://192.168.0.1/api/v2/pdus/16/sensors>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd" : "get",
  "token": "<token>"
}
```

RESPONSE

```
{ "ret-code": 0,
  "ret-msg": "",
  "data": {
    "pdus": [
      {
        "level": 16,
        "alive": 1,
        "sensors": [
          {
            "index": 1,
            "setting": {
              "type": 2,
              "name": "Temp=humid senso",
              "location": "sensor_location",
              "temp-alarm": 40,
              "temp-rising-alert": 0,
              "temp-low-alert": 90,
              "humid-alarm": 0,
              "humid-rising-alert": 0,
              "humid-low-alert": 0
            },
            "status": {
              "activation": 1,
              "connect-state": 1,
              "alarm-state": 0,
              "temp-measurement": 277,
              "temp-state": "Normal",
              "humid-measurement": 467,
              "humid-state": "Normal"
            }
          },
          {
            "index": 2,
            "setting": {
              "type": 2,
              "name": "Temp=humid senso",
              "location": "sensor_location",
              "temp-alarm": 40,
```

```

    "temp-rising-alert": 0,
    "temp-low-alert": 90,
    "humid-alarm": 0,
    "humid-rising-alert": 0,
    "humid-low-alert": 0
  },
  "status": {
    "activation": 1
    "connect-state": 1,
    "alarm-state": 0,
    "temp-measurement": 277,
    "temp-state": "Normal",
    "humid-measurement": 467,
    "humid-state": "Normal"
  }
}
]
}
]
}
}

```

Example S2) set temp alarm threshold of (alarm threshold(33.5 degree C) of sensor 1 on pdu level 16)

PATH: <https://192.168.0.3/api/v2/pdus>

HEADER:

Content-Type: application/json

BODY:

```
{
  "cmd": "set",
  "token": "<token>",
  "data": {
    "pdus": [
      {
        "level": 16,
        "sensors": [
          {
            "index": 1,
            "setting": {
              "temp-alarm": 33.5
            }
          }
        ]
      }
    ]
  }
}
```

RESPONSE:

```
{
  "ret-code": 0,
  "ret-msg": ""
}
```

The company reserves the right to modify product specifications without prior notice and assumes no responsibility for any error which may appear in this publication.

All brand names, logo and registered trademarks are properties of their respective owners.

Copyright 2025 Austin Hughes Electronics Ltd. All rights reserved.