NETWORK TECHNOLOGIES INCORPORATED
1275 Danner Dr Tel:330-562-7070
Aurora, OH 44202 Fax:330-562-1999
www.networktechinc.com

JSON API SUPPORT

Support has been built into the ENVIROMUX firmware to use JSON API to poll sensors using HTTP protocol like cURL command. To automate the interface between servers and the ENVIROMUX and provide data, the following instruction is provided.

Using cURL

Step 1: Get the session cookie by a HTTP POST request:

Get session cookie by sending 'username' and 'password' in POST body to endpoint '/goform/login'. We'll receive a JSON response with the 'sessionId' as a 'cookie' variable

Request:

curl -X POST -d "username=root&password=nti" http://192.168.3.216/goform/login

Response:

{ "success":"true", "cookie":"sessionId=cm9vdDpudGk6MTA=" }

Step 2: Get the sensor details using appropriate end point and by providing the session cookie in header.

Example with Endpoint of /json/get/applSens.json:

Request:

curl -v -H "Host: 192.168.3.216" -H "Cookie: sessionId=cm9vdDpudGk6MTA" http://192.168.3.216/json/get/applSens.json

Response:

Please refer to page 117 for example of the response format.

Using HTTP browser as an example

First login to the ENVIROMUX from the web browser. Then enter any of the listed requests in the URL bar to be provided with the desired information.

Note: Command syntax is case sensitive.

List of available API endpoints:

IPADDRESS/json/get/applSens.json - for Internal Sensors
IPADDRESS/json/get/appESens.json - for External Sensors
IPADDRESS/json/get/appDiginp.json - for Digital Inputs
IPADDRESS/json/get/appIpdev.json - for IP Devices
IPADDRESS/json/get/appIpsens.json - for IP Sensors
IPADDRESS/json/get/appEvents.json - for Events
IPADDRESS/json/get/appSmalerts.json - for Smart Alerts
IPADDRESS/json/get/appNetwork.json - to get Network Settings
IPADDRESS/json/get/appDevice.json - to get Device Settings
IPADDRESS/json/get/appAll.json - to get all of the above information in one API
IPADDRESS/json/get/appSNMPSens.json - to get SNMP sensor values

The tables on the following page provide definitions for the Type and Status numbers that will be provided. See page 3 for an example of a json response via HTTP.

Sensor ID Definitions:

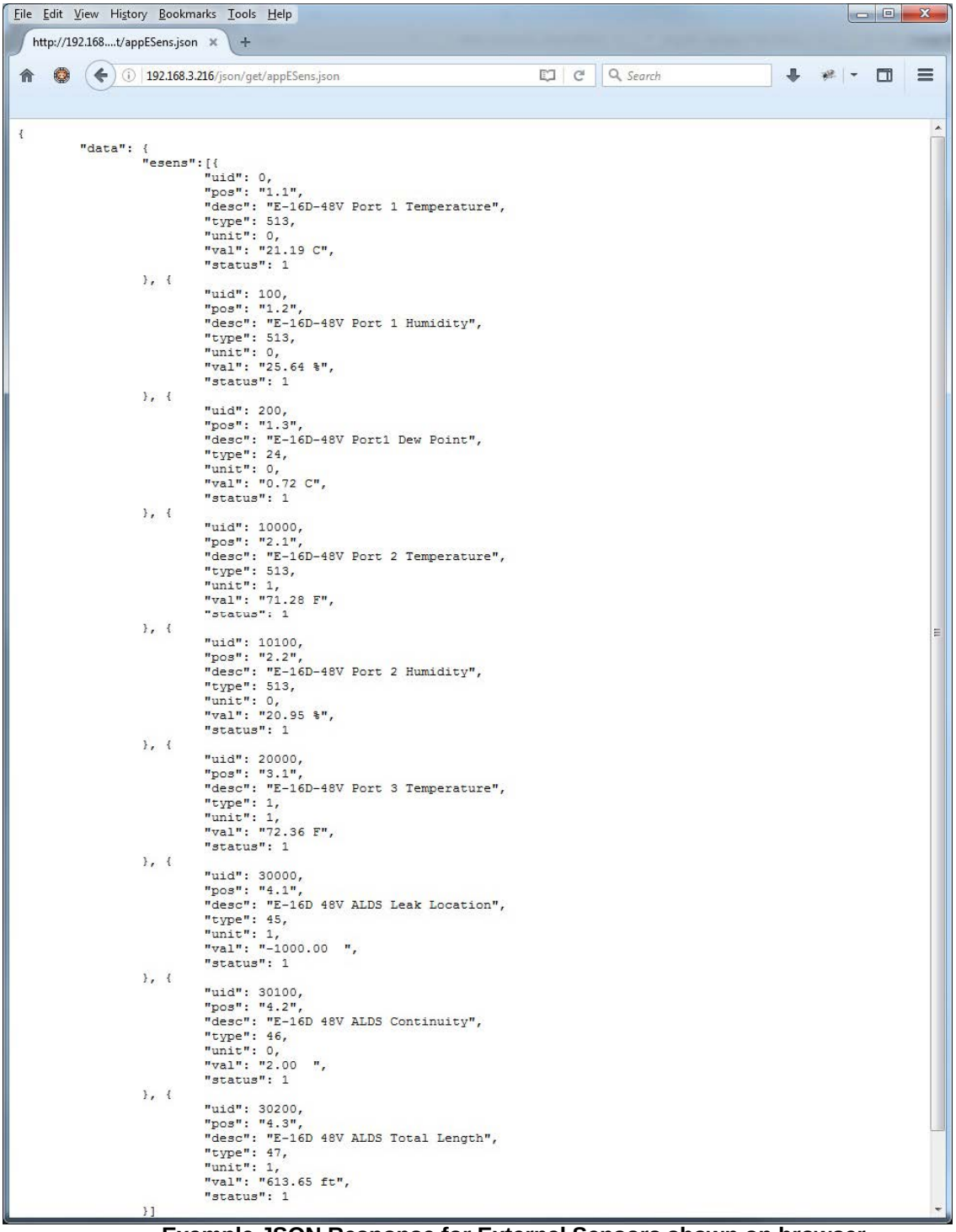
Sensor Type ID	Sensor Type	Sensor Type ID	Sensor Type	Sensor Type ID	Sensor Type
0	ID_UNDEFINED	20	ID_PING	42	ID_ACLM3_C
1	ID_TEMPERATURE	21	ID_NOT_RESPONDING	43	ID_ACLM3_W
2	ID_HUMIDITY	22	ID_LIGHT	44	ID_ACLM3_VAR
3	ID_POWER	23	ID_TEMPERATURE_EX	230	ID_POWER_SUPP
4	ID_LOW_VOLTAGE	24	ID_DEWPOINT	513	ID_TEMP_HUM
5	ID_CURRENT	25	ID_NLS	540	ID_TEMP_HUM_EX2
6	ID_ACLM_V	26	ID_TAC_DIO16	552	ID_TEMP_HUM_EX3
7	ID_ACLM_V_OF_P	27	ID_HUMIDITY_D	771	ID_POW_POW
8	ID_ACLM_P	28	ID_TEMPERATURE_EX2	1285	ID_CURR_CURR
9	ID_WATER	29	ID_TAC_DIP1	1028	ID_LOWV_LOWV
10	ID_SMOKE	30	ID_AIR_VELOCITY	1542	ID_ACLM_V_V
11	ID_VIBRATION	31	ID_DUST	1800	ID_ACLM_P_V
12	ID_MOTION	33	ID_RTD_TRANSMITTER	6913	ID_TEMP_HUM_D
13	ID_GLASS	35	ID_FREQUENCY	32769	ID_TEMP_COMBO
14	ID_DOOR	36	ID_AC_V	32796	ID_TEMP_COMBO_EX2
15	ID_KEYPAD	37	ID_AC_C	32808	ID_TEMP_COMBO_EX3
16	ID_PANIC_BUTTON	38	ID_DC_V	32770	ID_HUM_COMBO
17	ID_KEY_STATION	39	ID_DC_C	32767	ID_CUSTOM
18	ID_DRY_CONTACT	40	ID_TEMPERATURE_EX3	9767	ID_DCLM6
19	ID_DIG_INPUT	41	ID_ACLM3_V	9253	ID_ACLM3

Sensor Status ID	Sensor Status	Sensor Status ID	Sensor Status
0	STATUS_NOTCONNECTED	6	STATUS_DISCONNECTED
1	STATUS_NORMAL	7	STATUS_TAMPER_ALERT
2	STATUS_WARNING	8	STATUS_PREDIZZY
3	STATUS_ALERT	9	STATUS_DIZZY
4	STATUS_ACKNOWLEDGED	10	STATUS_IN_USE
5	STATUS_DISMISSED	11	STATUS_NOT_USED

HTTP Example:

Entered into the browser URL bar: **`<IP Address>/json/get/appESens.json`**

Response:




```
{
        "data": {
                "esens":[{
                        "uid": 0,
                        "pos": "1.1",
                        "desc": "E-16D-48V Port 1 Temperature",
                        "type": 513,
                        "unit": 0,
                        "val": "21.19 C",
                        "status": 1
                }, {
                        "uid": 100,
                        "pos": "1.2",
                        "desc": "E-16D-48V Port 1 Humidity",
                        "type": 513,
                        "unit": 0,
                        "val": "25.64 %",
                        "status": 1
                }, {
                        "uid": 200,
                        "pos": "1.3",
                        "desc": "E-16D-48V Port1 Dew Point",
                        "type": 24,
                        "unit": 0,
                        "val": "0.72 C",
                        "status": 1
                }, {
                        "uid": 10000,
                        "pos": "2.1",
                        "desc": "E-16D-48V Port 2 Temperature",
                        "type": 513,
                        "unit": 1,
                        "val": "71.28 F",
                        "status": 1
                }, {
                        "uid": 10100,
                        "pos": "2.2",
                        "desc": "E-16D-48V Port 2 Humidity",
                        "type": 513,
                        "unit": 0,
                        "val": "20.95 %",
                        "status": 1
                }, {
                        "uid": 20000,
                        "pos": "3.1",
                        "desc": "E-16D-48V Port 3 Temperature",
                        "type": 1,
                        "unit": 1,
                        "val": "72.36 F",
                        "status": 1
                }, {
                        "uid": 30000,
                        "pos": "4.1",
                        "desc": "E-16D 48V ALDS Leak Location",
                        "type": 45,
                        "unit": 1,
                        "val": "-1000.00  ",
                        "status": 1
                }, {
                        "uid": 30100,
                        "pos": "4.2",
                        "desc": "E-16D 48V ALDS Continuity",
                        "type": 46,
                        "unit": 0,
                        "val": "2.00  ",
                        "status": 1
                }, {
                        "uid": 30200,
                        "pos": "4.3",
                        "desc": "E-16D 48V ALDS Total Length",
                        "type": 47,
                        "unit": 1,
                        "val": "613.65 ft",
                        "status": 1
                }]
```

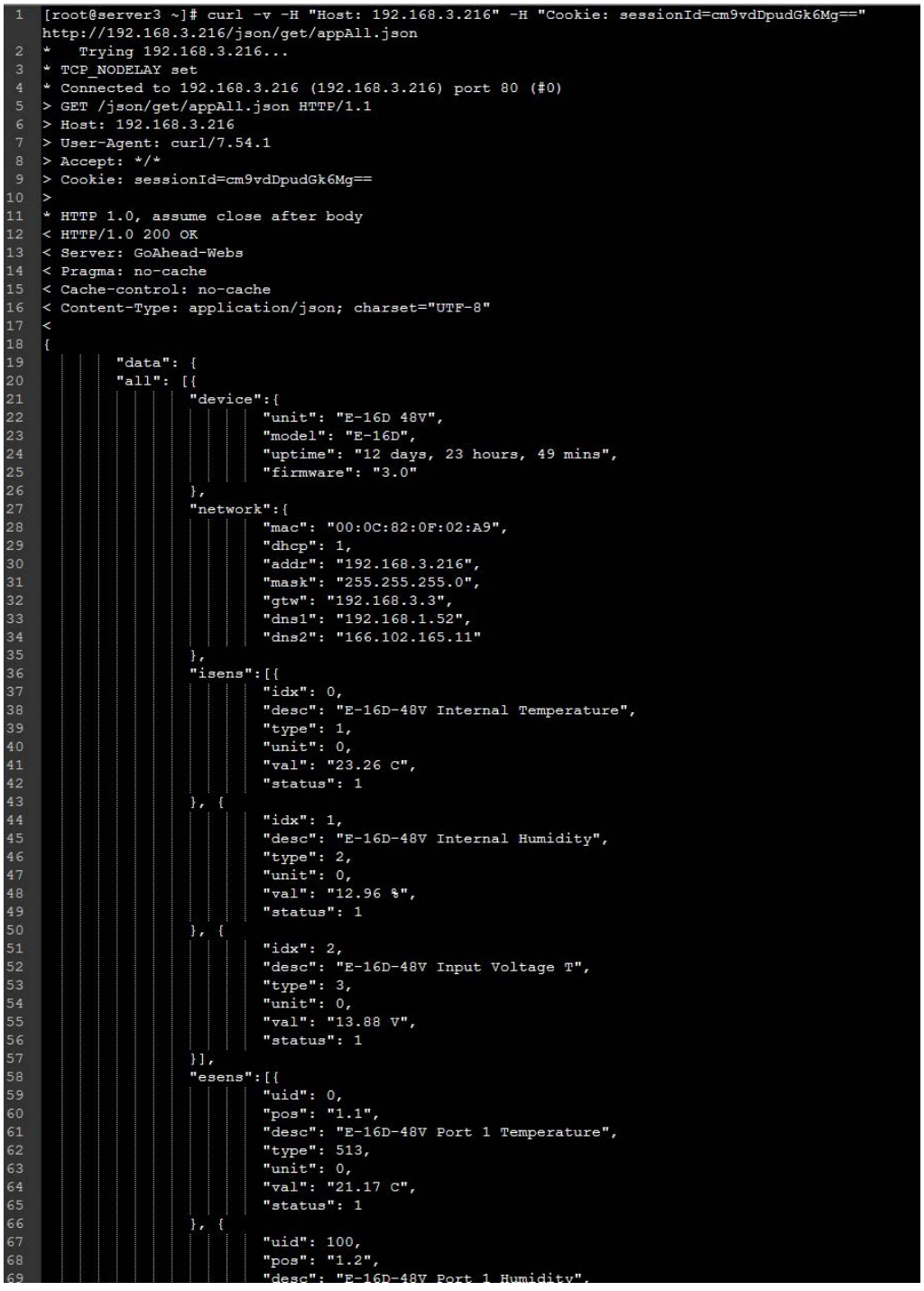

Example JSON Response for External Sensors shown on browser

cURL Example:

Entered at the command line after getting sessionId:

```
curl -v -H "Host: 192.168.3.216"  -H "Cookie: sessionId=cm9vdDpudGk6MTM"
http://192.168.3.216/json/get/appAll.json
```

Response:



```
[root@server3 ~]# curl -v -H "Host: 192.168.3.216" -H "Cookie: sessionId=cm9vdDpudGk6Mg=="
http://192.168.3.216/json/get/appAll.json
*   Trying 192.168.3.216...
* TCP_NODELAY set
* Connected to 192.168.3.216 (192.168.3.216) port 80 (#0)
> GET /json/get/appAll.json HTTP/1.1
> Host: 192.168.3.216
> User-Agent: curl/7.54.1
> Accept: */*
> Cookie: sessionId=cm9vdDpudGk6Mg==
>
* HTTP 1.0, assume close after body
< HTTP/1.0 200 OK
< Server: GoAhead-Webs
< Pragma: no-cache
< Cache-control: no-cache
< Content-Type: application/json; charset="UTF-8"
<
{
        "data": {
        "all": [{
                "device":{
                        "unit": "E-16D 48V",
                        "model": "E-16D",
                        "uptime": "12 days, 23 hours, 49 mins",
                        "firmware": "3.0"
                },
                "network":{
                        "mac": "00:0C:82:0F:02:A9",
                        "dhcp": 1,
                        "addr": "192.168.3.216",
                        "mask": "255.255.255.0",
                        "gtw": "192.168.3.3",
                        "dns1": "192.168.1.52",
                        "dns2": "166.102.165.11"
                },
                "isens":[{
                        "idx": 0,
                        "desc": "E-16D-48V Internal Temperature",
                        "type": 1,
                        "unit": 0,
                        "val": "23.26 C",
                        "status": 1
                }, {
                        "idx": 1,
                        "desc": "E-16D-48V Internal Humidity",
                        "type": 2,
                        "unit": 0,
                        "val": "12.96 %",
                        "status": 1
                }, {
                        "idx": 2,
                        "desc": "E-16D-48V Input Voltage T",
                        "type": 3,
                        "unit": 0,
                        "val": "13.88 V",
                        "status": 1
                }],
                "esens":[{
                        "uid": 0,
                        "pos": "1.1",
                        "desc": "E-16D-48V Port 1 Temperature",
                        "type": 513,
                        "unit": 0,
                        "val": "21.17 C",
                        "status": 1
                }, {
                        "uid": 100,
                        "pos": "1.2",
                        "desc": "E-16D-48V Port 1 Humidity",
```

Example JSON Response for all information using cURL

FAQ

Can I get individual sensor data or can I only get sensor data in bulk?

>>All sensors of the E-xD are available as listed in this document. There is an API to get all sensors data in bulk at once as well as individual API's for each sensor group.

Is there any technical limitation of the API's if we call each one of them in one second intervals, such that in one minute our code will call each sensor 60 times?

>>We recommend calling the API in 5 second intervals but there is no limitation to calling down to 1 second intervals. If calling every second we recommend to limit the use of other external services like SSH connections, SNMP calls, Modbus calls, having other web pages open at same time, etc.

What is the latency time of the API's?

>> Latency of API varies depending on usage and other factors like HTTPS use, HTTPS session resets, SNMP usage, terminal/SSH connections open, etc. Assuming limited use of other features, best case latency is typically under 300ms.

In such a scenario do you recommend having another architecture, other than using API's ?

>> If device is on the same local network, the response will be fast and suitable for real time updates. HTTP API's are a good choice for this application and it is the same API used by our management software. Other choices are SNMP and Modbus. Any one of these that suits your requirement can be used.

For the complete E-xD product manual with all features and functions, click here.

sman154-07 3/23/22