NETIO

Networked power sockets

NETIO integration guide

1) AV drivers business model

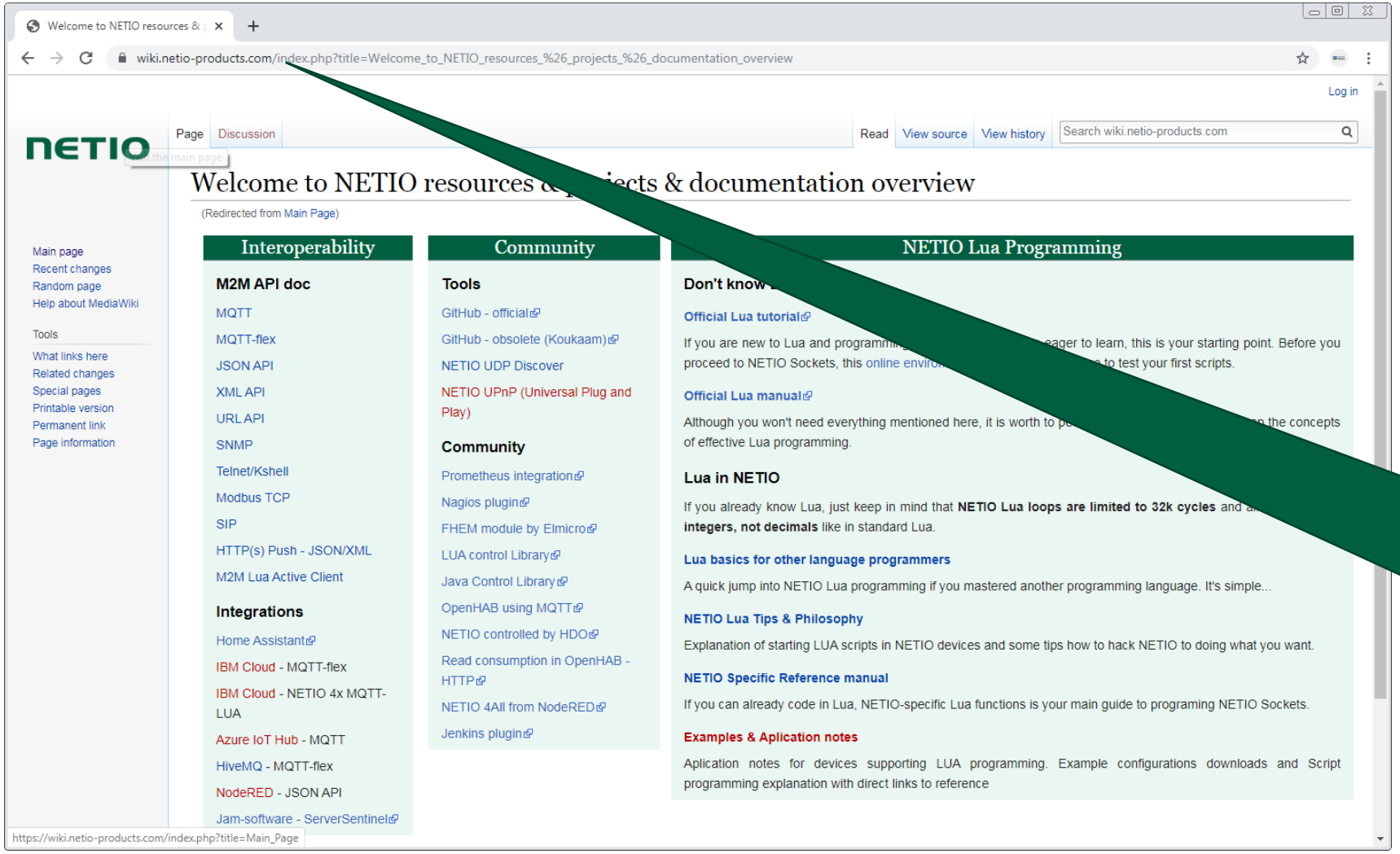
2) JSON (HTTP)

3) MQTT-flex

4) NETIO cloud

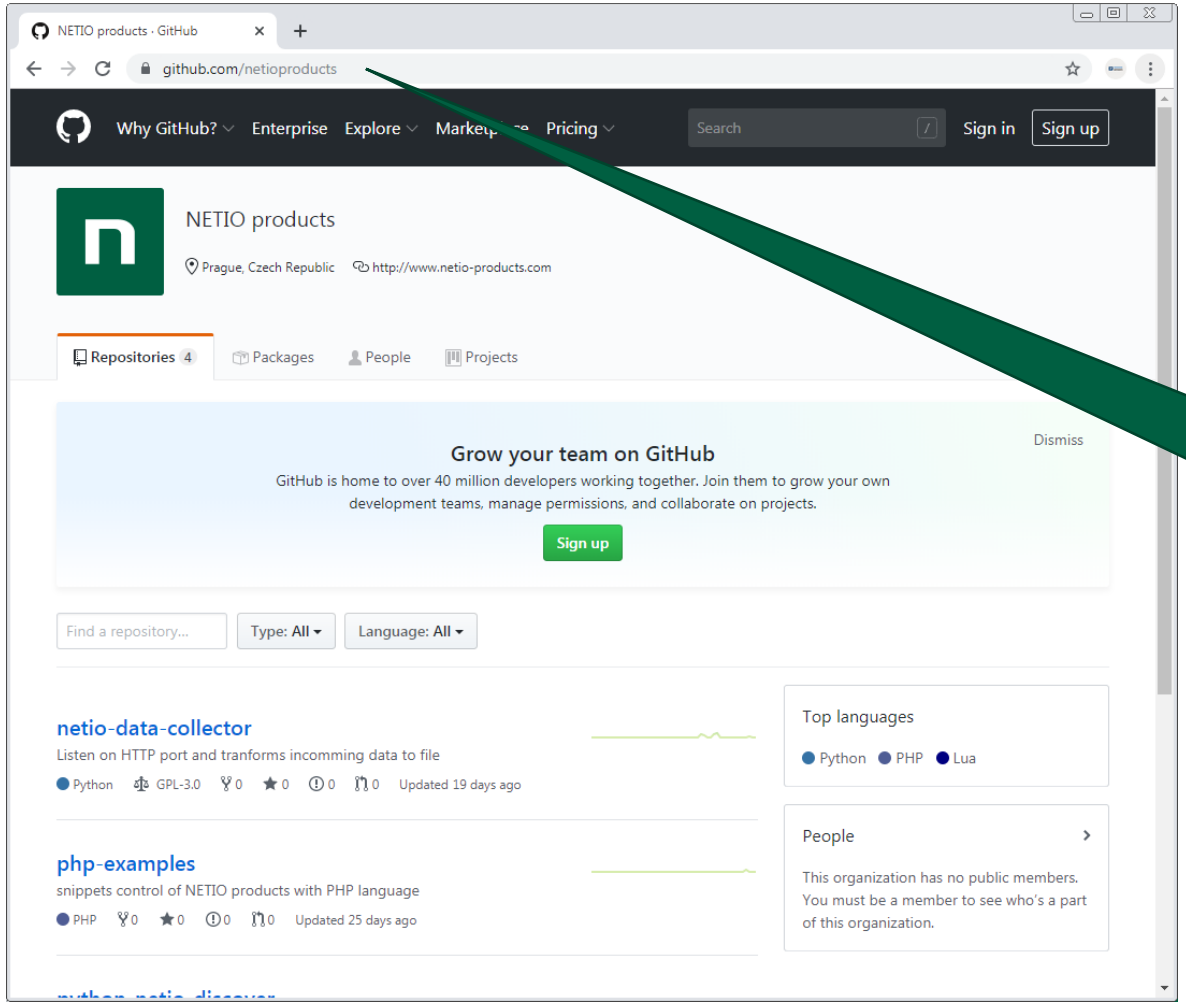
www.netio-products.com

Wiki.netio-products.com



NETIO Wiki for programmers

https://github.com/netioproducts



GitHub with code examples for programmers

NETIO UDP discover: How to find a device on the LAN

Application Notes (ANxx) | Page | NETIO PowerDIN 4Pz | NETIO pro | powerdin-4pz.netio-products.com | NETIO UDP Discover - wiki.netio-

wiki.netio-products.com/index.php?title=NETIO_UDP_Discover

Log in

NETIO

Page | Discussion | Read | View source | View history | Search wiki.netio-products.com

Main page
Recent changes
Random page
Help about MediaWiki

Tools
What links here
Related changes
Special pages
Printable version
Permanent link
Page information

NETIO UDP Discover

NETIO Discover uses UDP broadcast transmission.

- NETIO GitHub: **Python Netio Discover Class**
  Automatically detect all interfaces and sends UDP discover to search ne... on connected interfaces.
- NETIO UDP Discover - **Java based Multiplatform - JAR**

Send Discover message:

Destination: Broadcast (eg. 255.255.255.0)
Protocol: UDP
Destination Port: 62387
Data(HEX): 01ec00

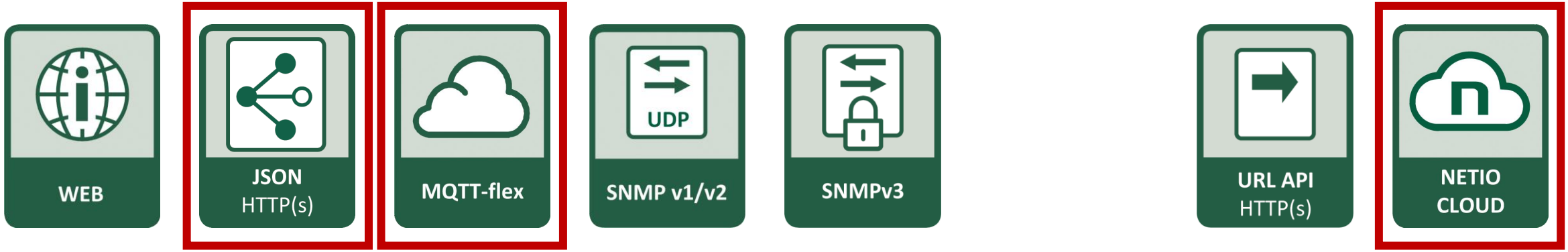
Receive Discover response:

NETIO devices response with UDP broadcast message with destination port 62386. So listen on port 62386 and parse
are sent as datastream with flags inside. If you pass the end of known field types, ignore rest of data in datastream.

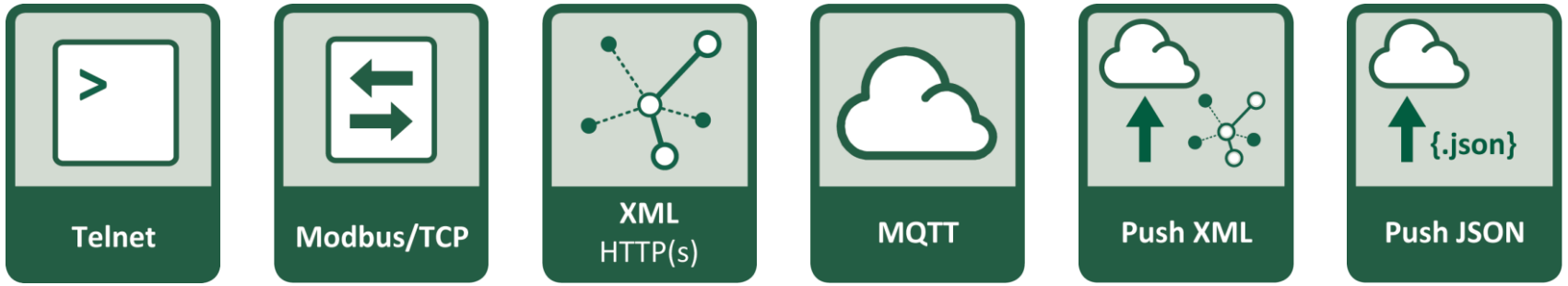
Listen protocol: UDP
Listen on port: 62386
Data(HEX):

Let customer enter the IP address of the device or find it on LAN from your SW…

NETIO Open API



Supported, but not recommended protocols:



(1)
3RD party AV DRIVERS business models

NETIO
Networked power sockets

AV drivers from NETIO

Neets

ELAN®

Control4™

HS

CRESTRON®

domotz

RTI

savant
NOW YOU CAN

Coming soon:

cue

QSC

Extron®

SYMCON

SKAARHOJ
Engineering. Multimedia. Consulting.

NETIO AV drivers recommendation: Free / Paid version

Free of charge driver	Paid driver
▪ Define IP address, port, username / password by JSON ▪ Write 0 / 1 state of all outputs by JSON ▪ Toggle state of all outputs by JSON ▪ Read current states of all outputs (feedback) (some of drivers only) ▪ Max 10 devices in the system supported (some of drivers only)	▪ Device discover (search for the devices) ▪ Support HTTPs (security) ▪ Read states of all outputs (if not in free version) ▪ Unlimited amount of devices supported in one system ▪ Reading power metering data (V, A, W, Wh, TPF per output) ▪ Generating Wh averages (kWh consumption in last 1h / 4h / 24h for example)

Choose the right Protocols for integration

NETIO
Networked power sockets

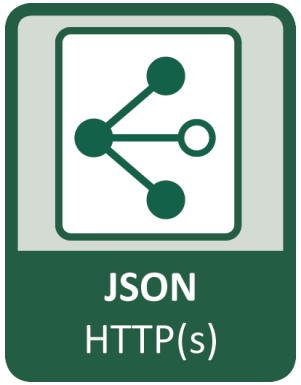
NETIO recommended protocols for integration

Inside building	Cloud oriented integration
▪ On the same LAN ▪ We recommend **JSON over HTTP(s)** ▪ Long-term stable are http based protocols ▪ http can be upgraded to https when needed ▪ We prefer JSON than XML ▪ All NETIO products has JSON as default enabled protocol. ▪ There is one JSON structure for all NETIO devices.	▪ Active protocol needed to connect from location behind NAT / Firewall. ▪ We recommend **MQTT(s)** ▪ SSL / HTTPs requires ▪ **MQTT-flex** recommended (flex = NETIO extension of MQTT protocol to customize data structure / period, target server etc.. ▪ Periodical HTTP push is also cloud oriented protocol option.

(2)
JSON HTTP(s)

NETIO
Networked power sockets

NETIO protocols: JSON



1) JSON is the **default enabled** protocol in all NETIO devices.

2) **/netio.json** is the R/W file with JSON structure. Check AN21 + download JSON documentation.

3) There is device MAC address as unique identificator Agent / SerialNumber in the JSON structure.

4) All standard NETIO products contains Agent / VendorID = 0, this can be modified up on request.

5) Device identification: How many outputs is per device is defined in the tag Agent / NumOutputs.
You can analyze it from the JSON structure but there can be 1 or 8 outputs.

6) NETIO DEFAULT:
HTTP port 80 like the device web.
Username/password for writing is “**netio**” / “**netio**”.

MAC address is printed on each device

NETIO protocols: JSON

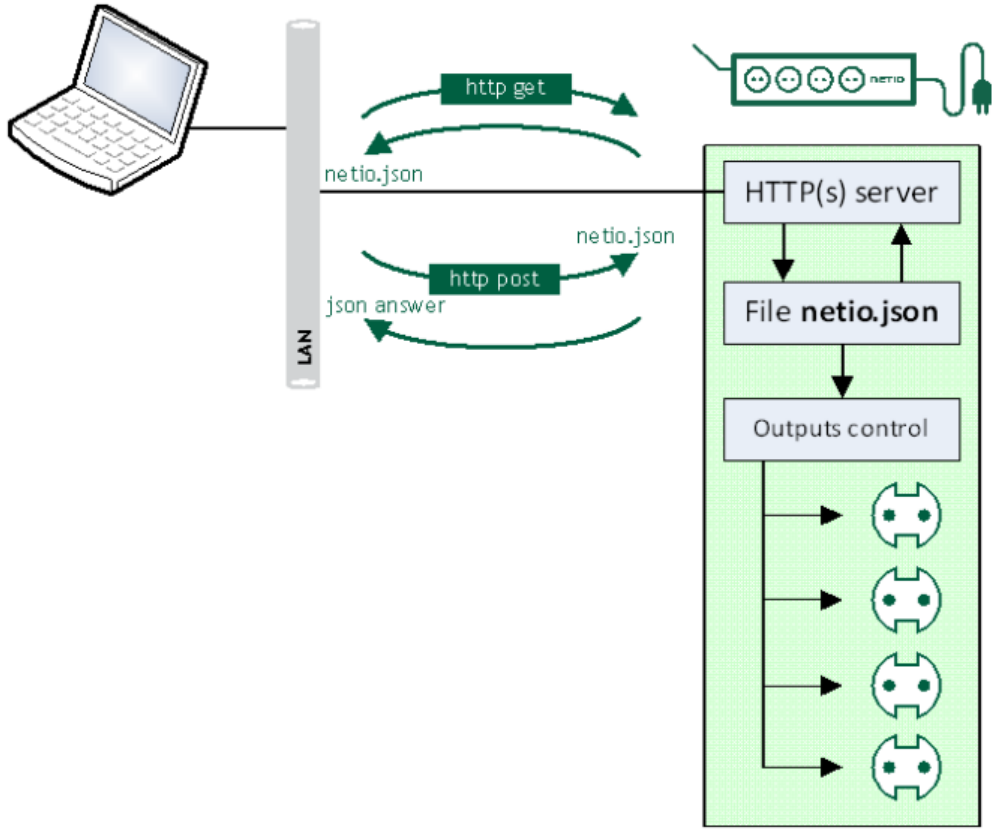
Application Notes (ANxx) | Page | NETIO PowerDIN 4Pz | NETIO pro | powerdin-4pz.netio-products.com | view-source:powerdin-4pz.netio-

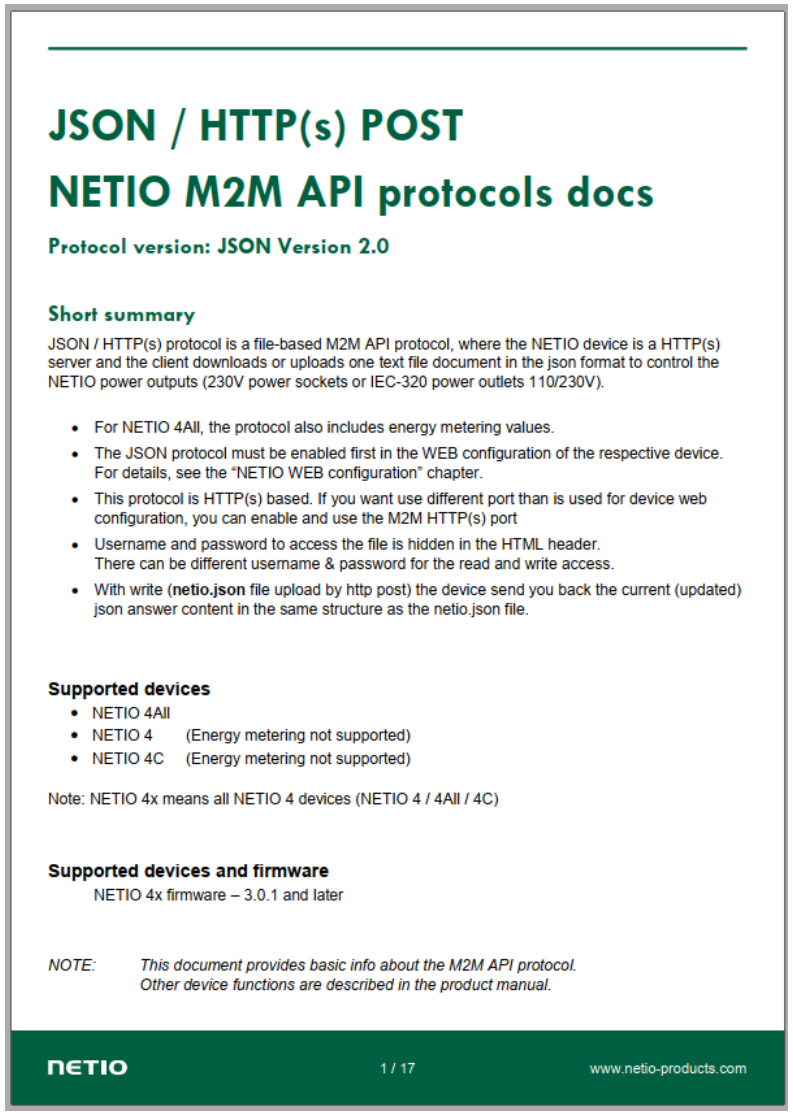
Not secure | view-source:powerdin-4pz.netio-products.com:22888/netio.json


```
{
"Agent":{"Model":"4PZ","DeviceName":"powerdin-4pz","MAC":"24:A4:2C:39:67:17","SerialNumber":"24A42C396717","JSONVer":"2.3","Time":"2020-10-03T23:21:47+01:00","Uptime":147,"Version":"2.5.4","OemID":400,"VendorID":0,"NumOutputs":4,"NumInputs":2},
"GlobalMeasure":{"Voltage":238,"TotalCurrent":0,"TotalLoad":0,"TotalEnergy":25,"OverallPowerFactor":0.00,"Frequency":50.08,"Phase":0.00,"EnergyStart":"1970-01-01T00:00:00+01:00"},
"Outputs":[
{"ID":1,"Name":"Power output 1","State":1,"Action":6,"Delay":2020,"Current":0,"PowerFactor":1.00,"Phase":0.00,"Energy":24,"ReverseEnergy":0,"Load":0},
{"ID":2,"Name":"Power output 2","State":1,"Action":6,"Delay":2020,"Current":0,"PowerFactor":1.00,"Phase":0.00,"Energy":0,"ReverseEnergy":0,"Load":0},
{"ID":3,"Name":"Free Contact 3","State":1,"Action":6,"Delay":2020},
{"ID":4,"Name":"Free Contact 4","State":0,"Action":6,"Delay":2020}
],
"Inputs":[
{"ID":1,"Name":"Intput 1","State":1,"S0Counter":55},
{"ID":2,"Name":"Intput 2","State":0,"S0Counter":1}
]}
```

Download the JSON manual in PDF

NETIO web >> Download >>
Download NETIO Open API





JSON / HTTP(s) POST

NETIO M2M API protocols docs

Protocol version: JSON Version 2.0

Short summary

JSON / HTTP(s) protocol is a file-based M2M API protocol, where the NETIO device is a HTTP(s) server and the client downloads or uploads one text file document in the json format to control the NETIO power outputs (230V power sockets or IEC-320 power outlets 110/230V).

- For NETIO 4All, the protocol also includes energy metering values.
- The JSON protocol must be enabled first in the WEB configuration of the respective device. For details, see the "NETIO WEB configuration" chapter.
- This protocol is HTTP(s) based. If you want use different port than is used for device web configuration, you can enable and use the M2M HTTP(s) port
- Username and password to access the file is hidden in the HTML header. There can be different username & password for the read and write access.
- With write (**netio.json** file upload by http post) the device send you back the current (updated) json answer content in the same structure as the netio.json file.

Supported devices

- NETIO 4All
- NETIO 4 (Energy metering not supported)
- NETIO 4C (Energy metering not supported)

Note: NETIO 4x means all NETIO 4 devices (NETIO 4 / 4All / 4C)

Supported devices and firmware

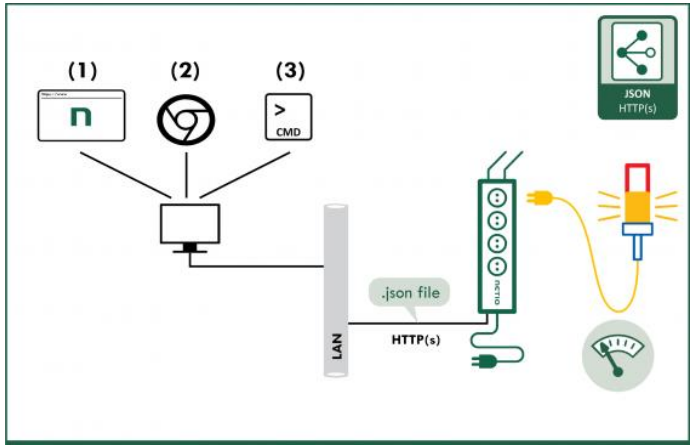
NETIO 4x firmware – 3.0.1 and later

NOTE: This document provides basic info about the M2M API protocol. Other device functions are described in the product manual.

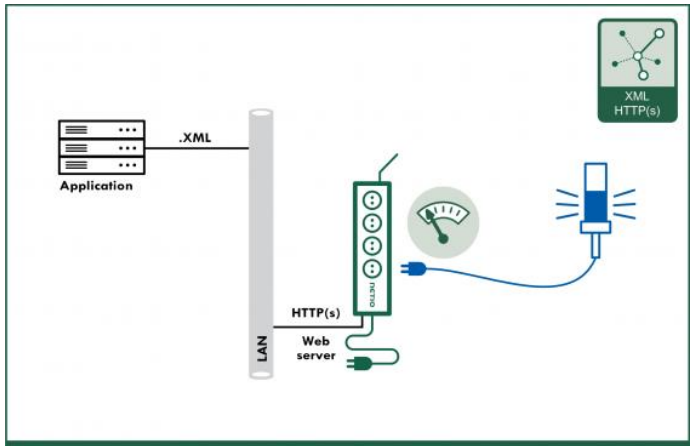
NETIO 1 / 17 www.netio-products.com

Start in 10 minutes: AN20 & AN21

- Check the (Web >> Support >> Application Notes) **AN20** / **AN21** on NETIO website.
- Here is shown how to control output from device's web interface & test the XML or JSON file.
- You can do this test even without the device on your table – use the online demo of each device.



NETIO AN21: JSON HTTP(s) protocol to control NETIO 110/230V power sockets (3x REST API)



NETIO AN20: XML HTTP(s) protocol to control NETIO power sockets/outlets

Video how to test XML / JSON from any NETIO device

JSON over HTTP: Tips & tricks

1) Enable your customers to be able modify
   - **IP address:port**
   - **Username** (default “netio”)
   - **Password** (default “netio”)

2) There can be different amount of outputs (different devices)

3) Not all outputs have to be metering energy (PowerDIN 4PZ for example)

4) Feel free to use “Toggle” action, not only 0/1 state. Keep in your mind, with using the action tag, the state tag will be ignored.

5) Restart time can be defined if you need.

To be done (ideal model)

1) Implement **Device search** on LAN is documented: https://wiki.netio-products.com/index.php?title=NETIO_UDP_Discover

2) Show the Device name (or MAC address) for device identification.

3) Predefine HTTP port = **80** + "**netio**" / "**netio**" as JSON default (let user possibility to change it).

4) Analyze amount of outputs, which are measured.
Do not expect it will be 4 outputs all the time.

5) Read State parameter, but write to Action to be able make short pulses / Toggle output.

6) Show the output numbers (ID) + names to make it easy for customers (enable users to edit text name of the output).

7) Be prepared to HTTPs sooner or later

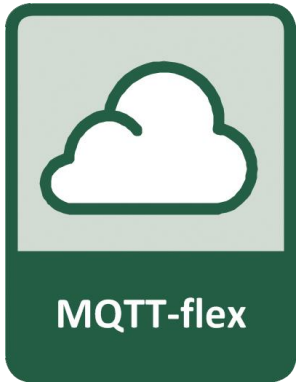
JSON Integration summary

	Your software	
LAN (UDP) **Discover**	No	
User definable **IP:port**	Yes	
Device identification	MAC + Device name (editable)	
Protocol:	JSON over HTTP	
Security options	https (not supported today)	
User definable **Username / Psw**	Yes	
Read status of all outputs (analyzed from JSON)	Yes	
Write status(action) to all outputs	Actions supported	
Energy metering	Yes (all supported ports)	

(3)
MQTT-flex
3rd party Cloud applications

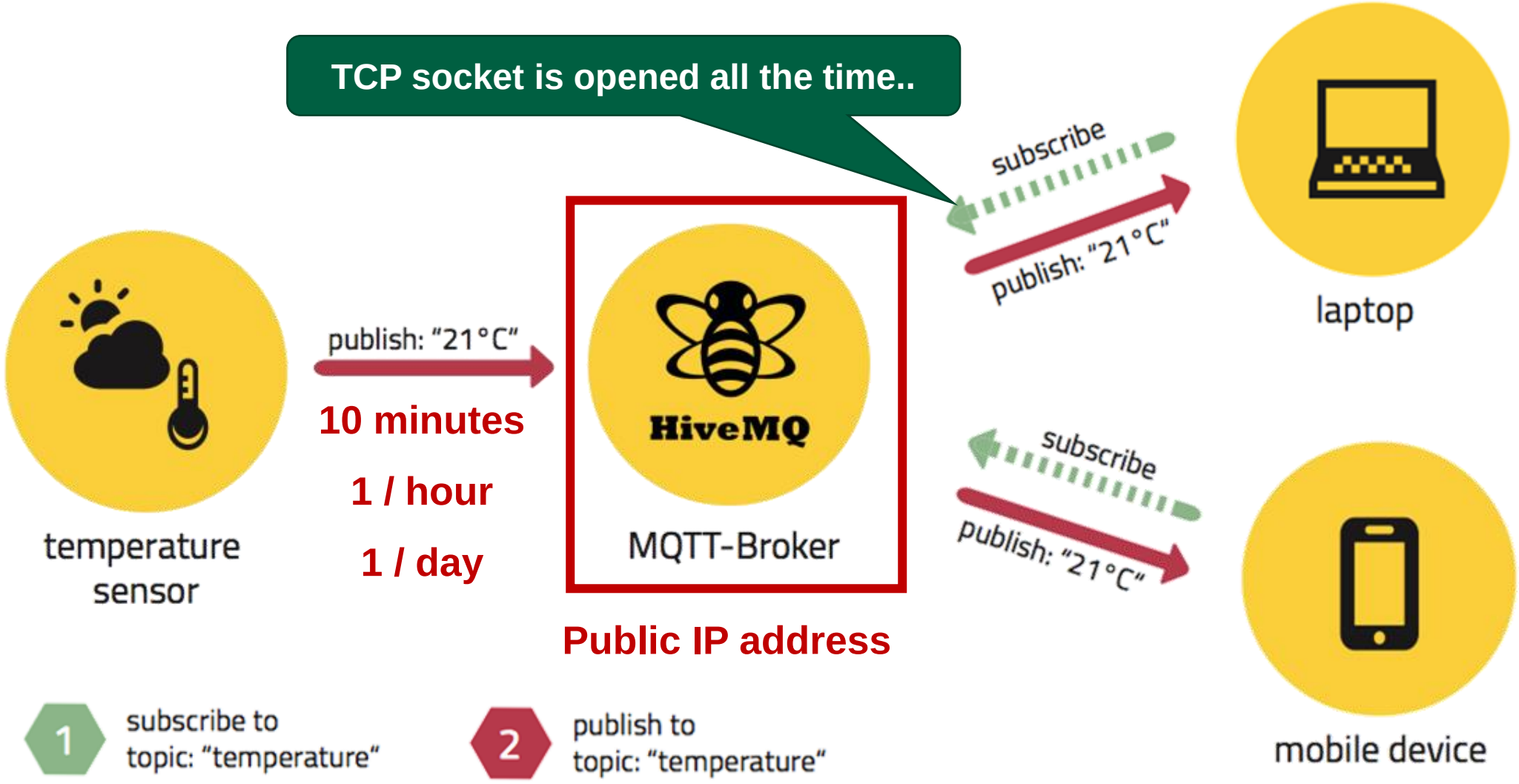
NETIO
Networked power sockets

NETIO protocols: MQTT-flex

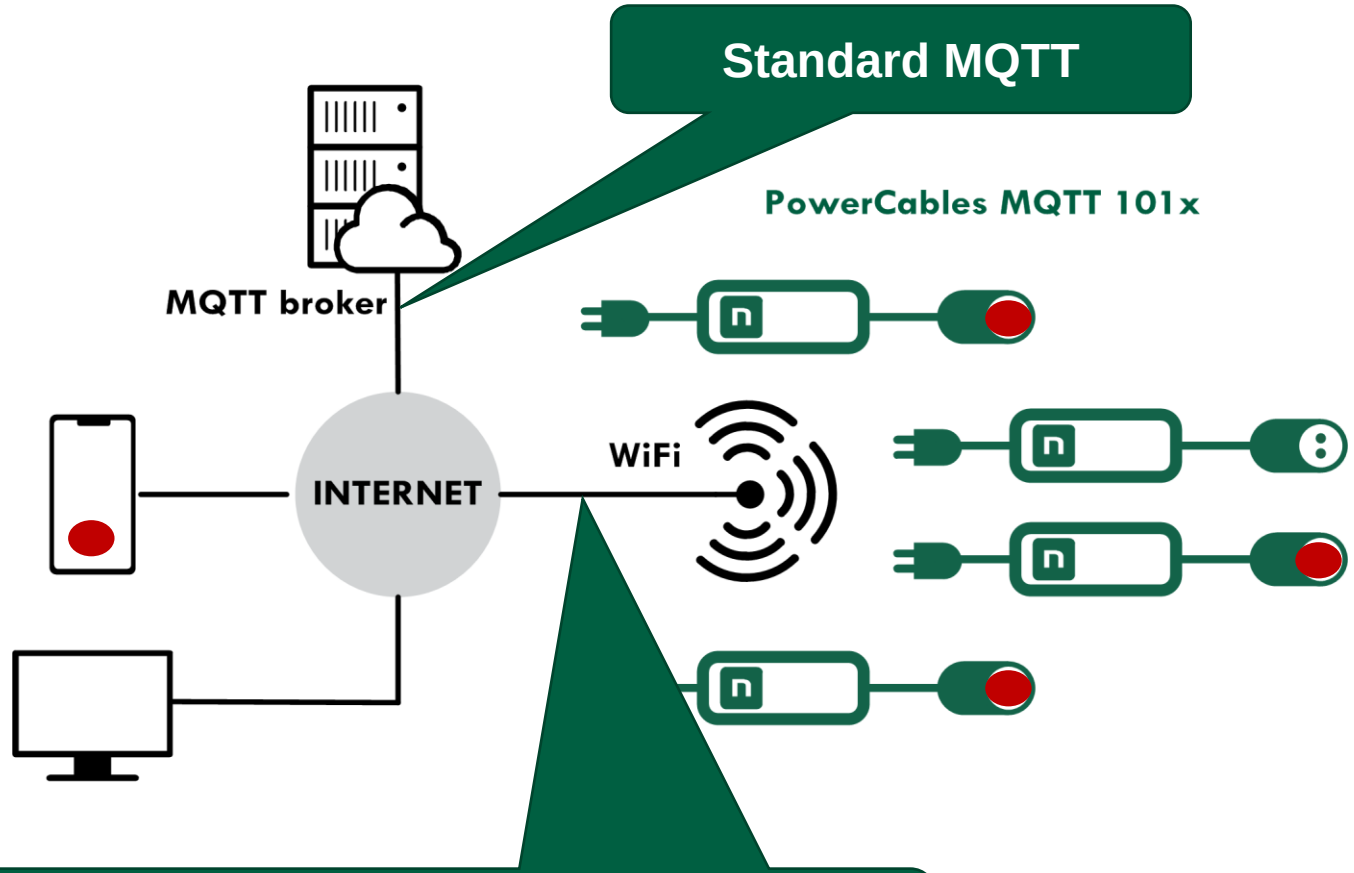


1) MQTT-flex protocol is supported in all NETIO devices (except Linux based PowerPDU 4C).
2) MQTT is ideal protocol for cloud oriented communication with NAT on the way.
3) MQTT-flex is **standard MQTT protocol** (ports 1883 or 8883) with easy to configure detailed conditions.
4) Customer’s MQTT-flex configuration can be uploaded to the device with the configuration file.

How works the MQTT

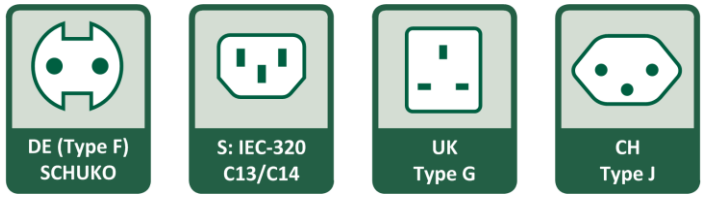


What is MQTT(-flex) by NETIO useful for?



flex = Configurable MQTT communication
(period, format, structure)

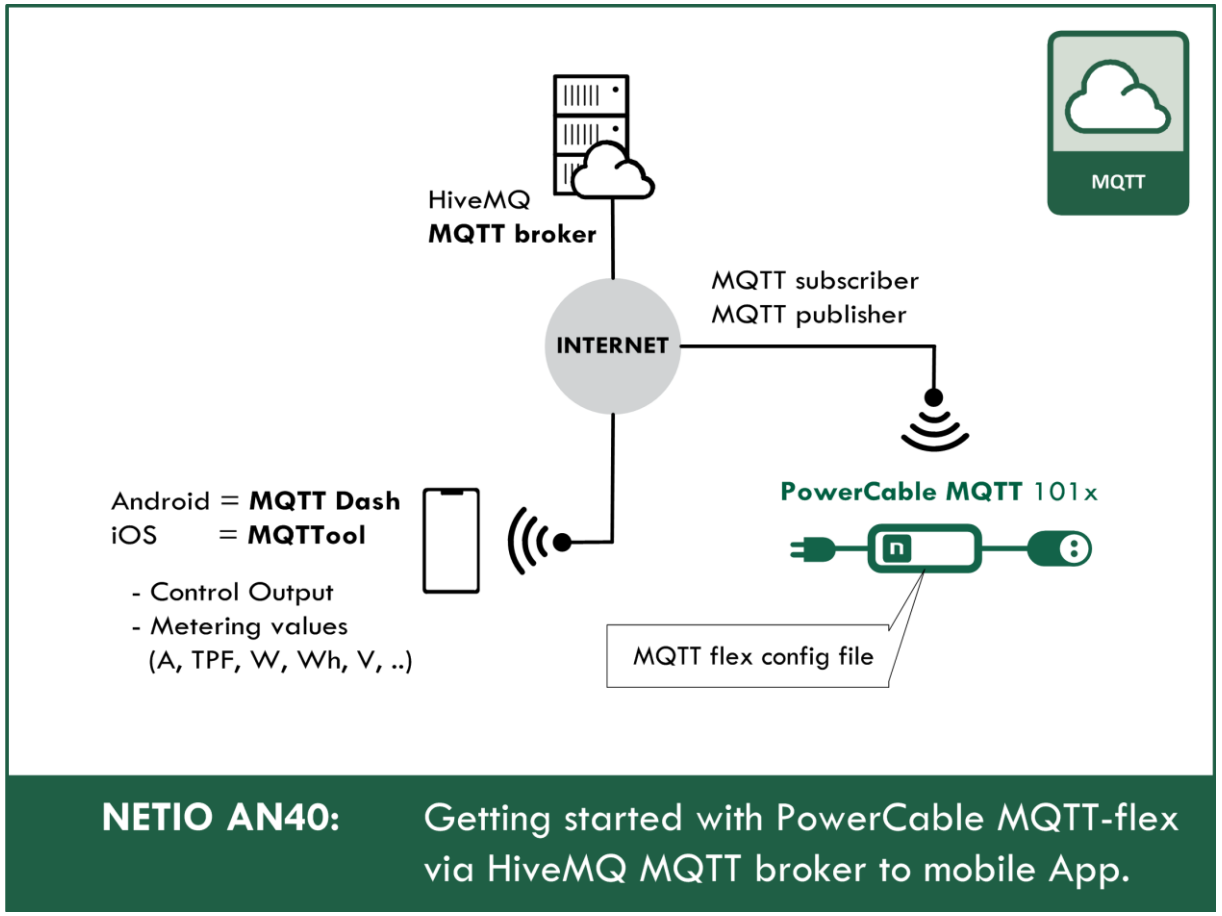
1) Remote **Power switching**

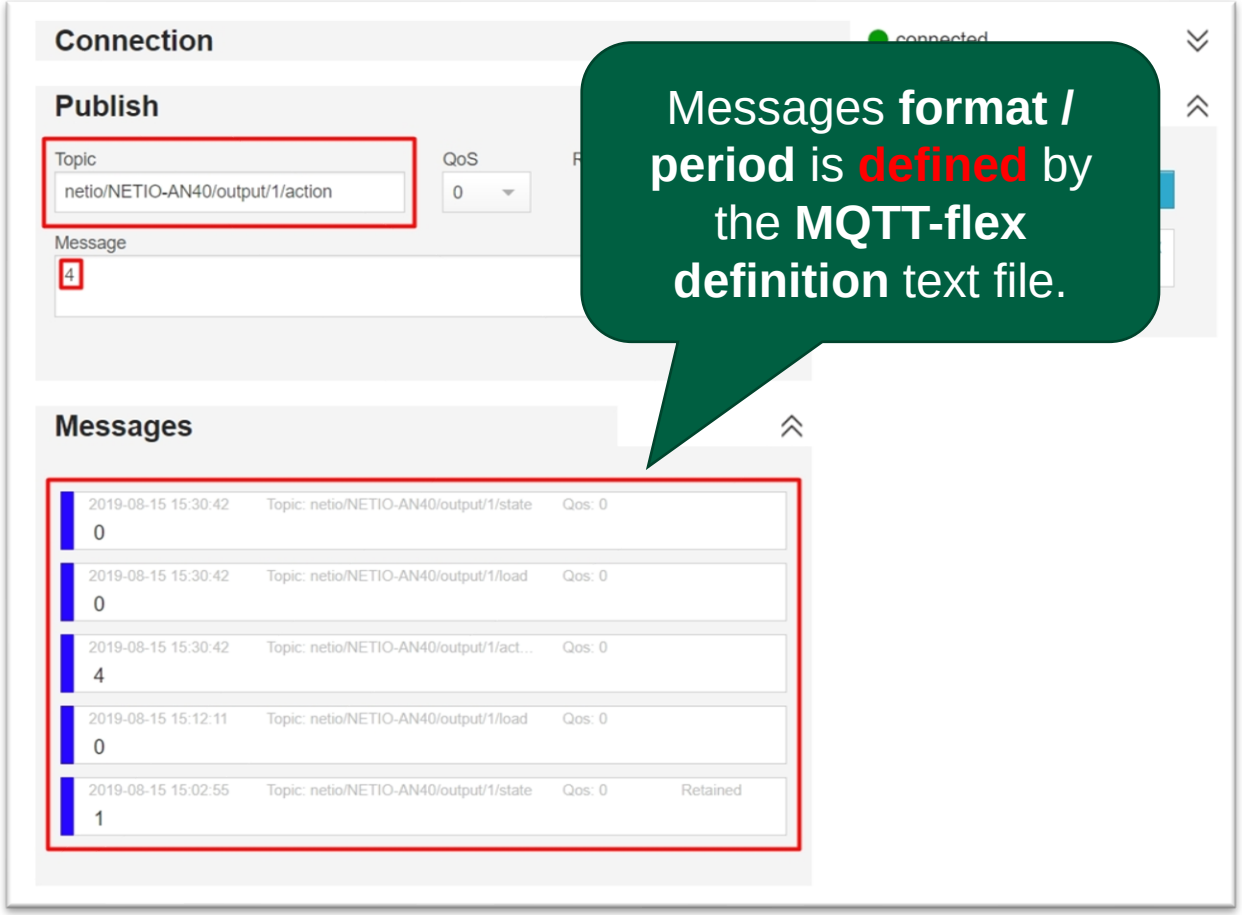


2) Remote **Power metering**

From any application running in the cloud or portal.

Check the details in AN40

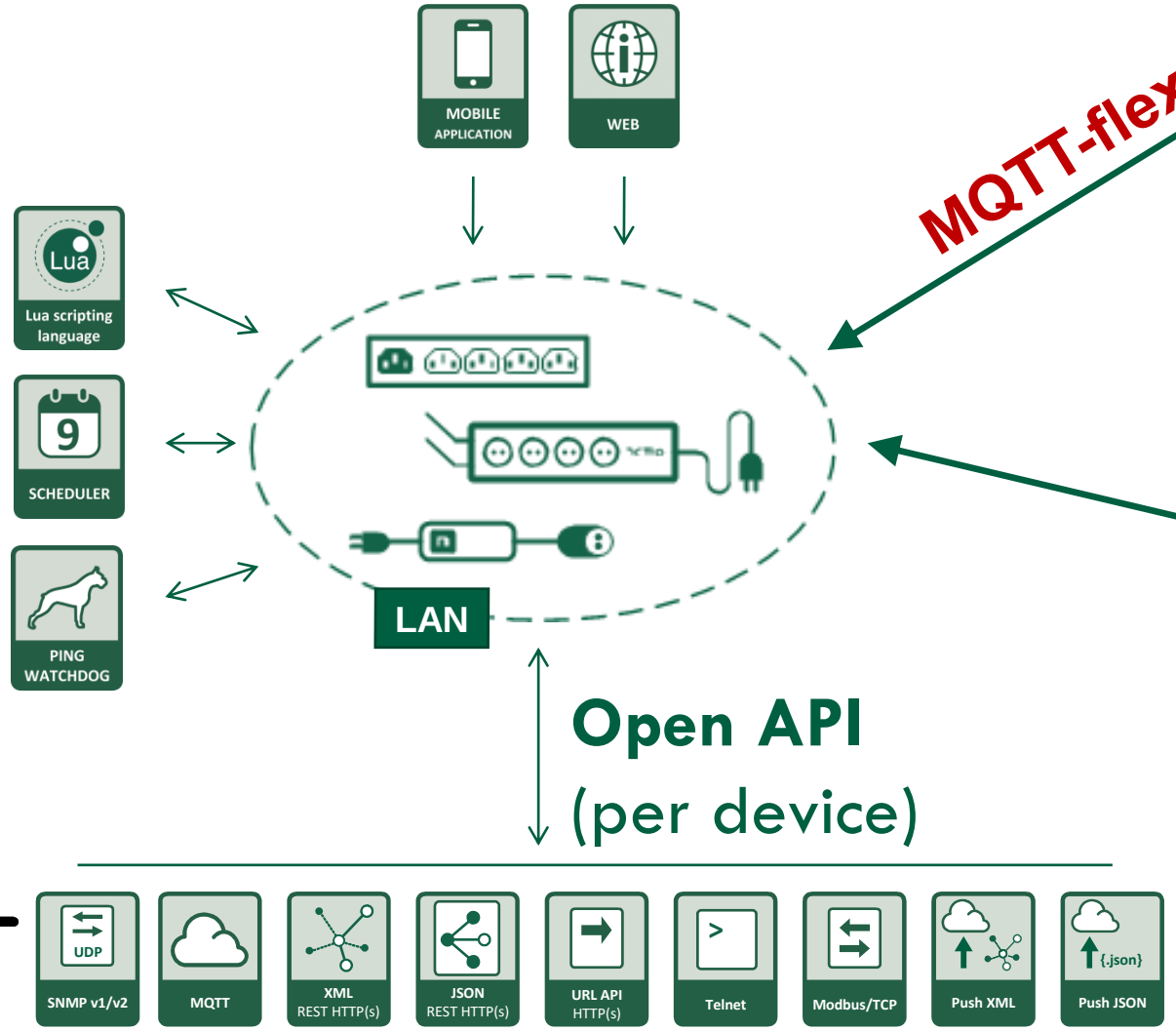




MQTT-flex details on **wiki.netio-products.com**

HiveMQ broker is used again for testing the communication.

NETIO clouds strategy



MQTT-flex

Your own cloud

OPEN API

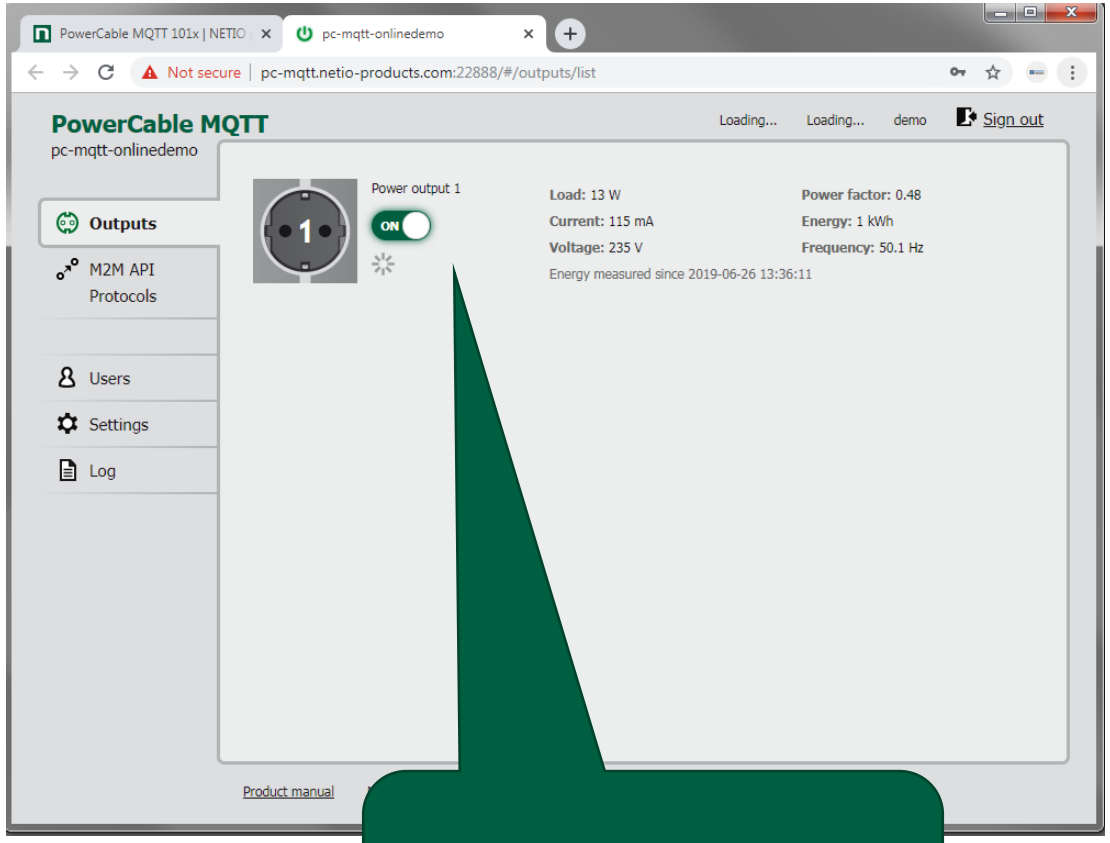
- MQTT-flex
- HTTP based protocols
- 1 device = 1 connection
- 9 devices = 9 connections

NETIO CLOUD

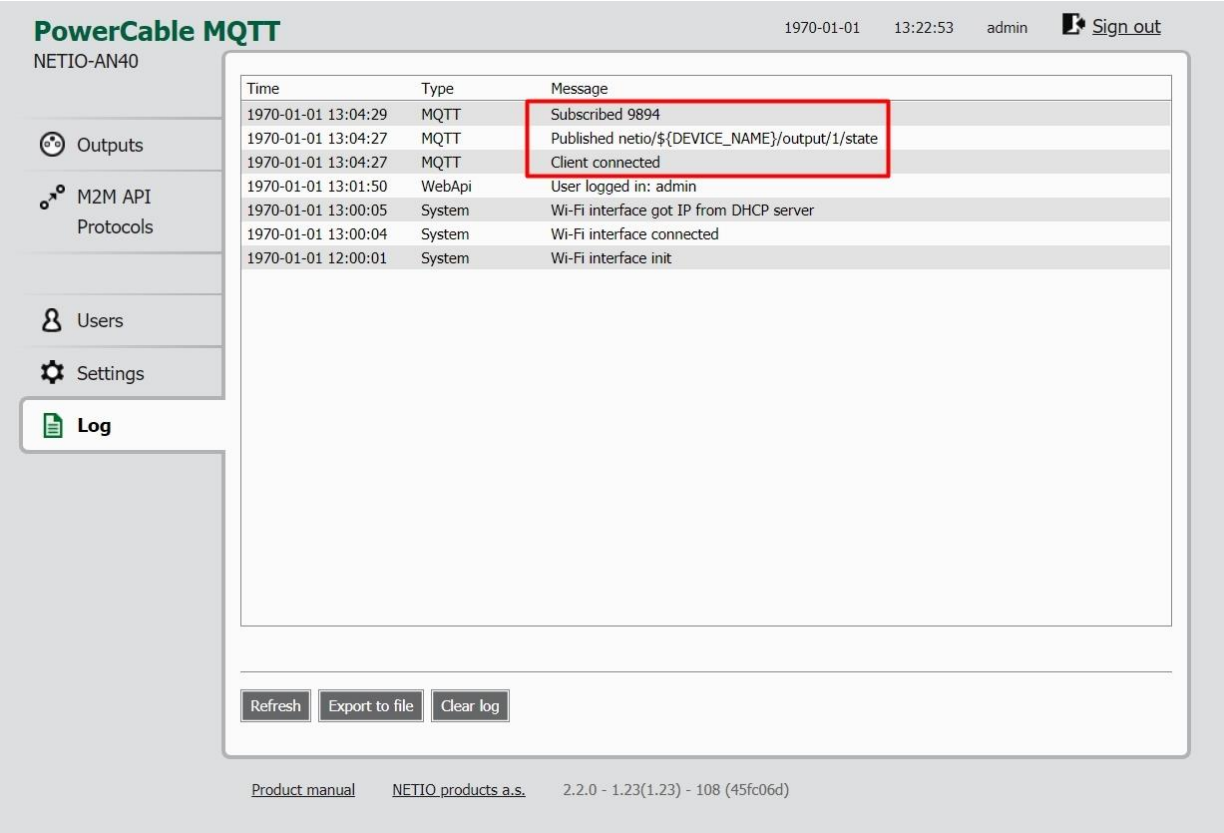
NETIO CLOUD

- 3 years „free of charge“
- Later: 5-15€ / device / year
- Remote On/Off only
- WEB + Mobile access

Device web interface



Click to control the power output.



Logfile is showing status of communication.

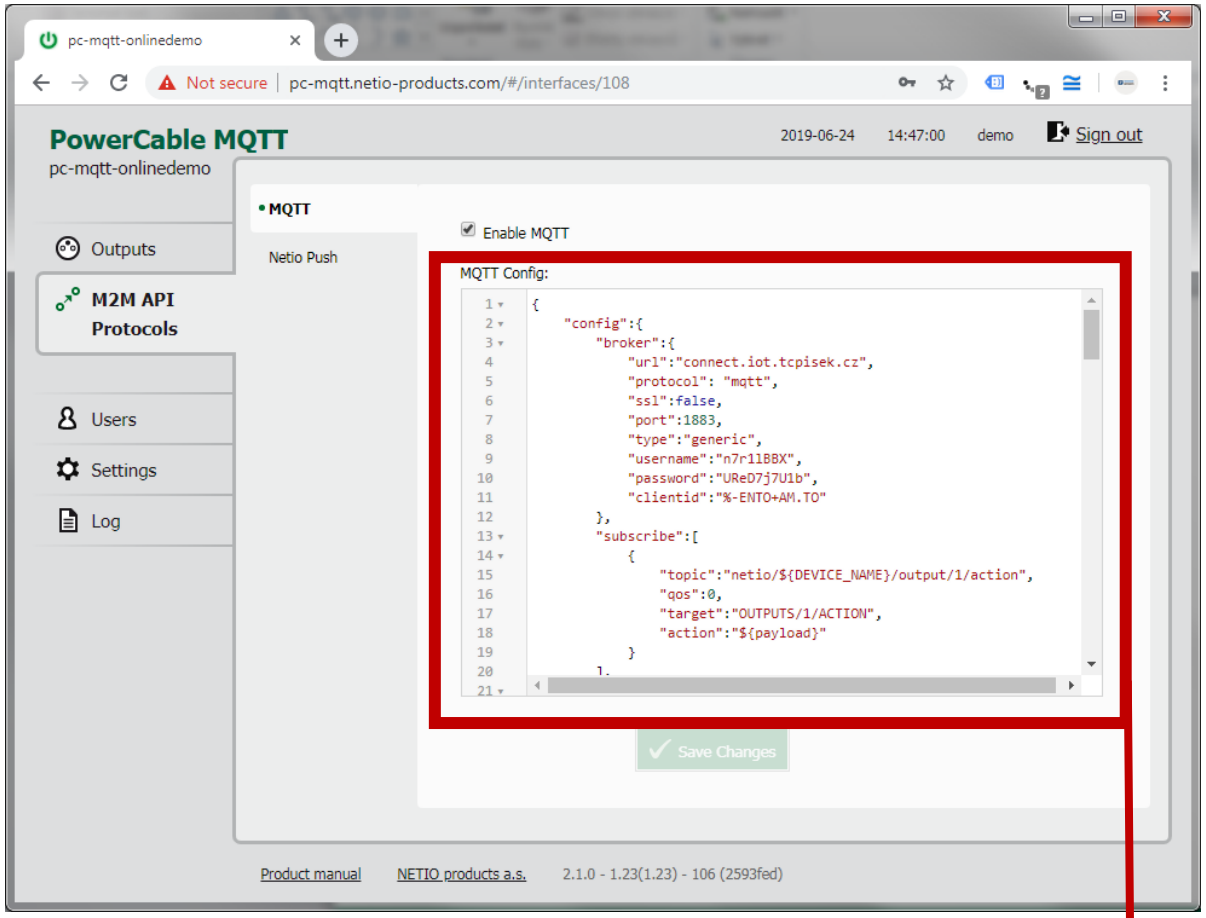
MQTT-flex (configuration)

- Cloud oriented protocol
- Standard TCP/IP port 1883 / 8883 (ssl)
- MQTT broker is standard service

- All NETIO devices (except PowerPDU 4C)

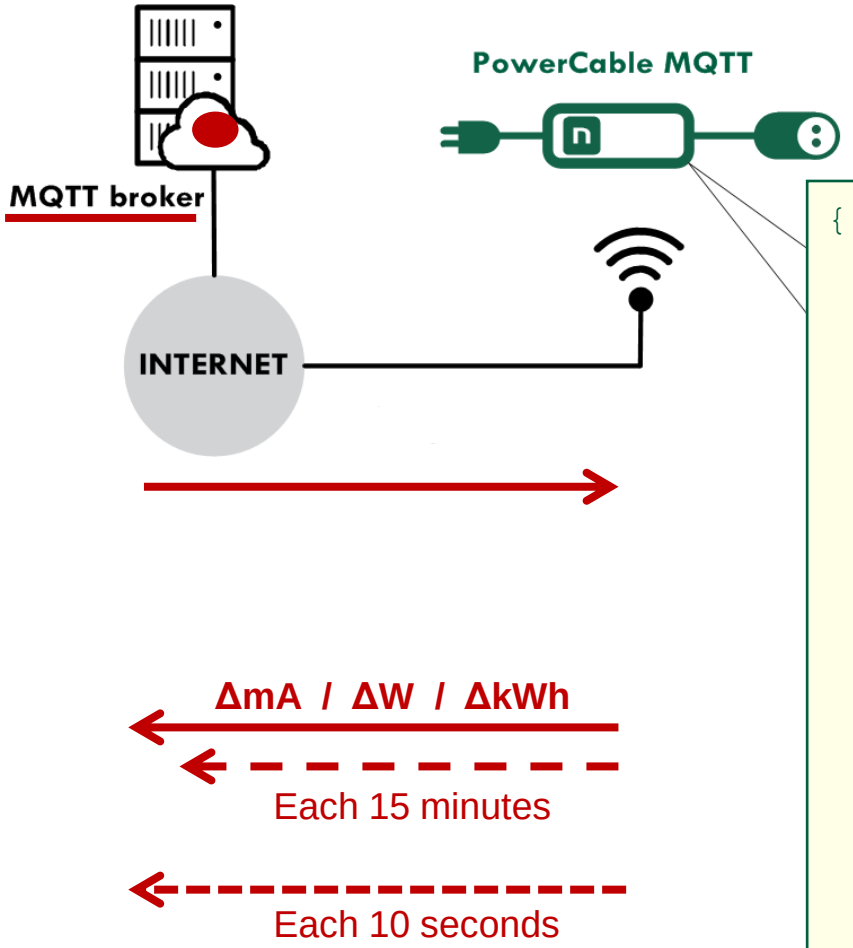
- **Editable** topics structure (MAC / IP / …)
- **Editable** payload structure
- **Configurable** data publishing conditions

- **Standard MQTT** data from broker's perspective



- **PowerCable MQTT** 101x
- MQTT-flex **config file** (text file)

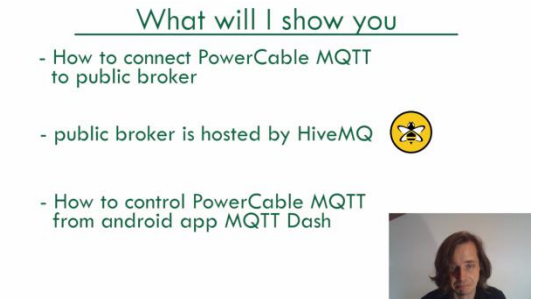
MQTT-flex (cloud) communication

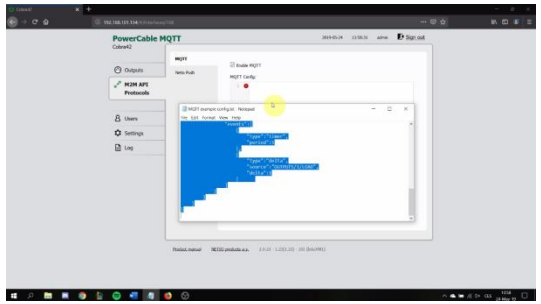


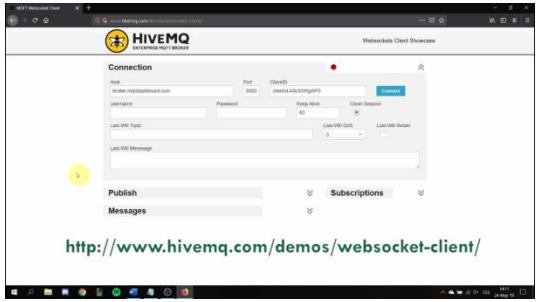

```
{
    "config":{
        "broker":{
            <broker definitions>
        },
        "subscribe":[
            {
                <subscribe topic 1 definitions>
            }
        ],
        "publish":[
            {
                <publish topic 1 definitions: events periodic / delta>
            },
            {
                <publish topic 2 definitions: events periodic>
            }
        ]
    }
}
```

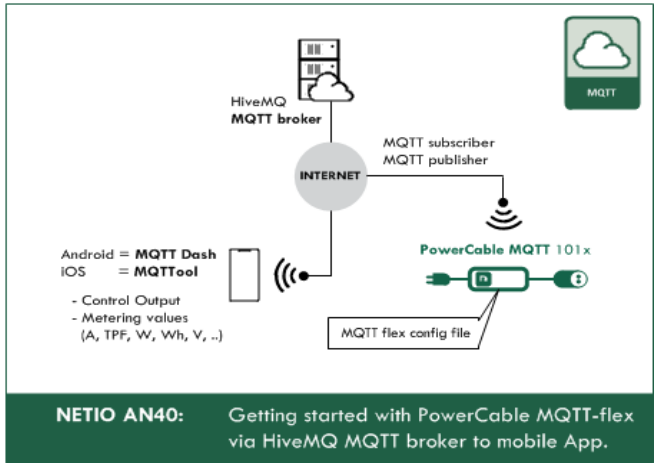
MQTT-flex: Next step

MQTT flex details on **wiki.netio-products.com**









AN40 PowerCable MQTT – getting started – HiveMQ

AN40 video with step by step guide from the device to the mobile app.

(4)
NETIO Cloud
Integration of all devices in one user account

NETIO
Networked power sockets

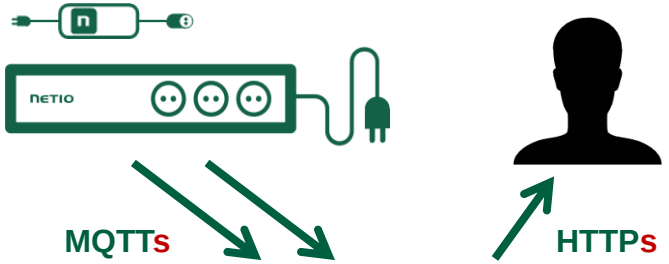
NETIO CLOUD

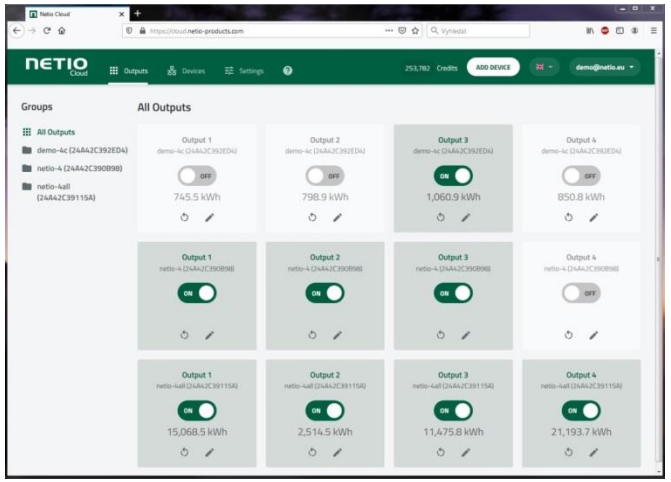
1 web screen to control 100 sockets

NETIO CLOUD service

NETIO Cloud is **1 screen** to control **100 outputs** from anywhere.

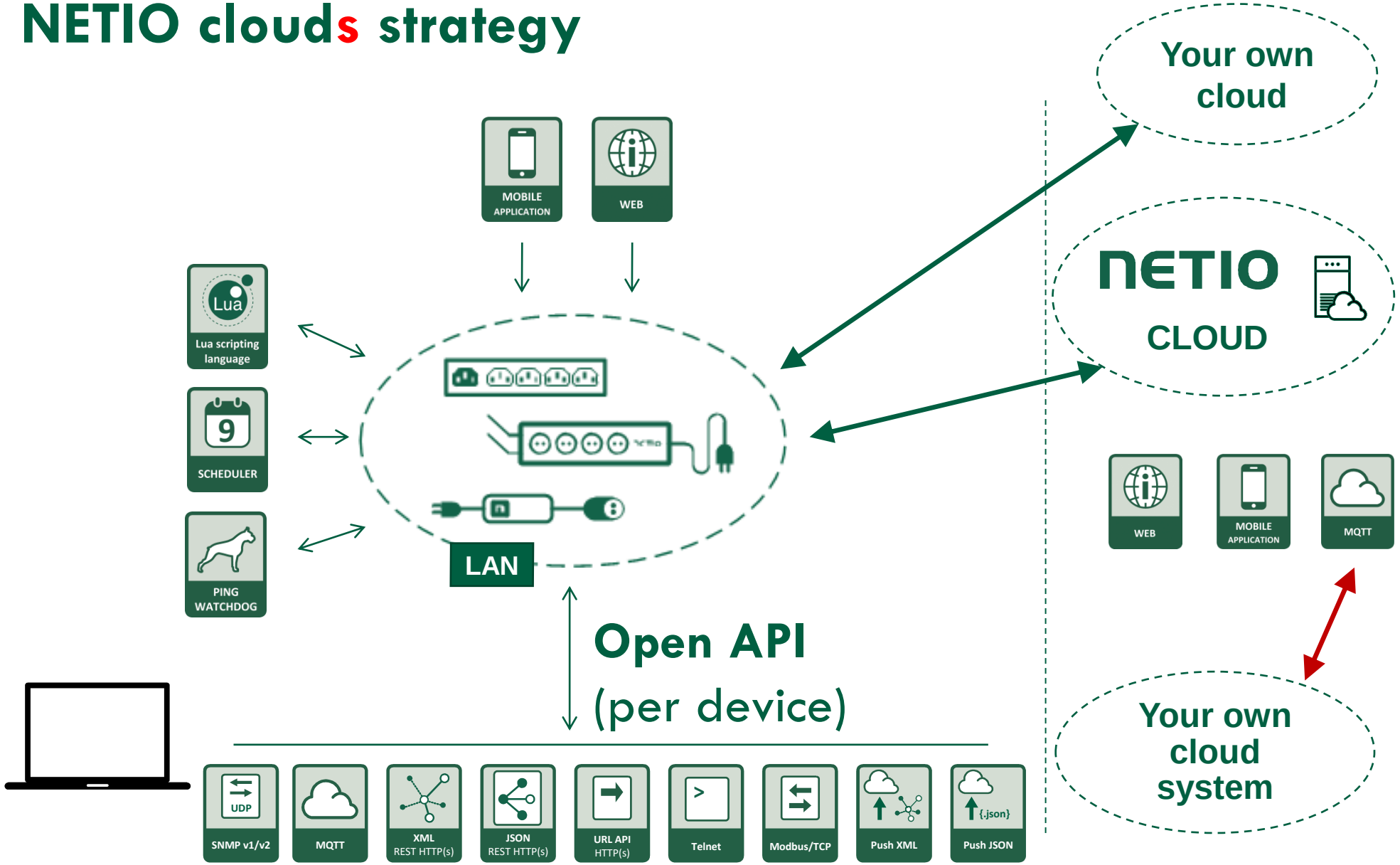
- Secured and **encrypted communication:** Devices >> Cloud >> User.
- Supported by all NETIO devices.
- Can be user in parallel to local software (**backup channel**).
- Open API for NETIO Cloud service is available.
- Data effective (in case of LTE connectivity)





- NETIO Cloud is **free for first 3 years**
- NETIO Cloud is very cheap service. (**5€ / device / year**)
- Can be used, don't have to be used..
  NETIO Cloud is just an option..

NETIO clouds strategy



OPEN API

- MQTT-flex
- HTTP based protocols
- 10 devices = 10 connections

NETIO CLOUD

- 3 years „free of charge“
- Later: 5-15€ / device / year
- Remote On/Off only
- WEB + Mobile access
- Node-RED installation

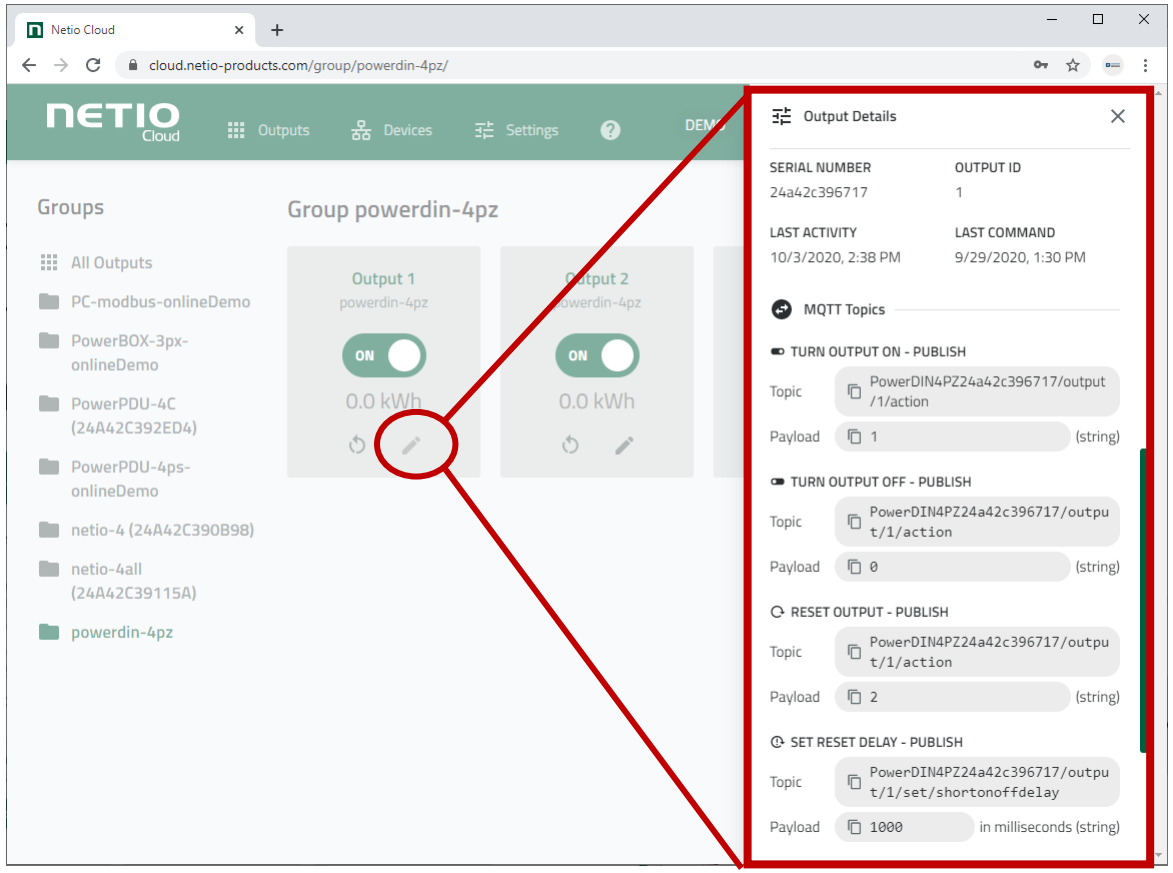
CLOUD connection

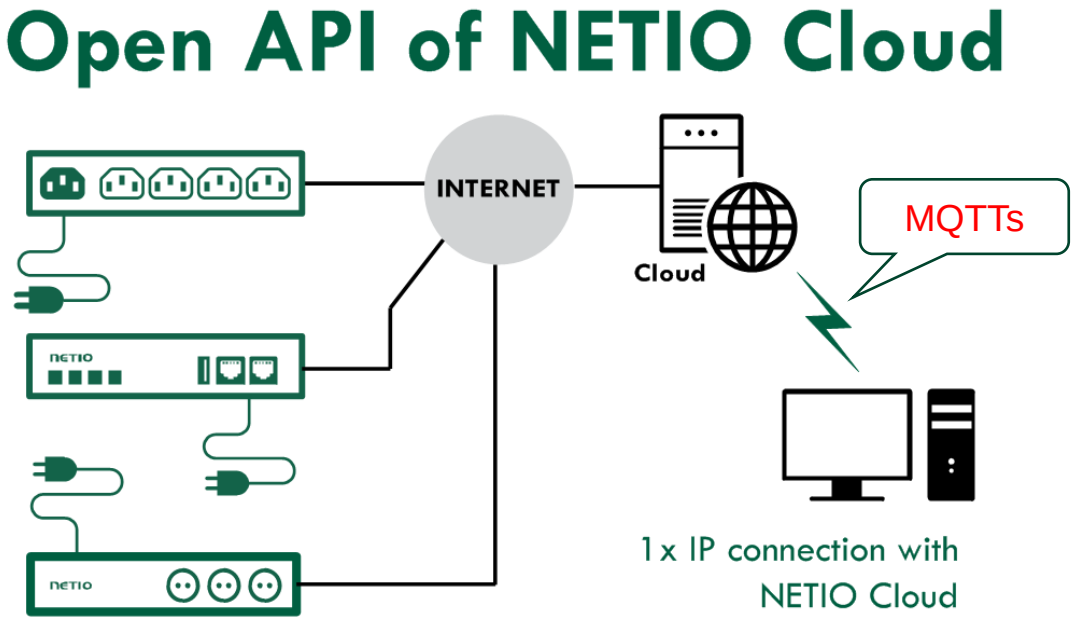
- 10 devices = **1** connection
- Open API to the cloud

Open API of NETIO Cloud service

NETIO Cloud is 1 screen to control 100 outputs from anywhere.

Not only from the screen, you can control them also from the Cloud Open API.





THANK YOU
FOR YOUR ATTENTION

NETIO
Networked power sockets