iNode LAN Central

Instruction manual

1. About iNode

We would like to introduce you to the family of iNode devices operating in Bluetooth Low Energy ® technology. We will show you that BLE is not only tags for finding lost keys or location tags, but something more.

Our devices can do this and more:

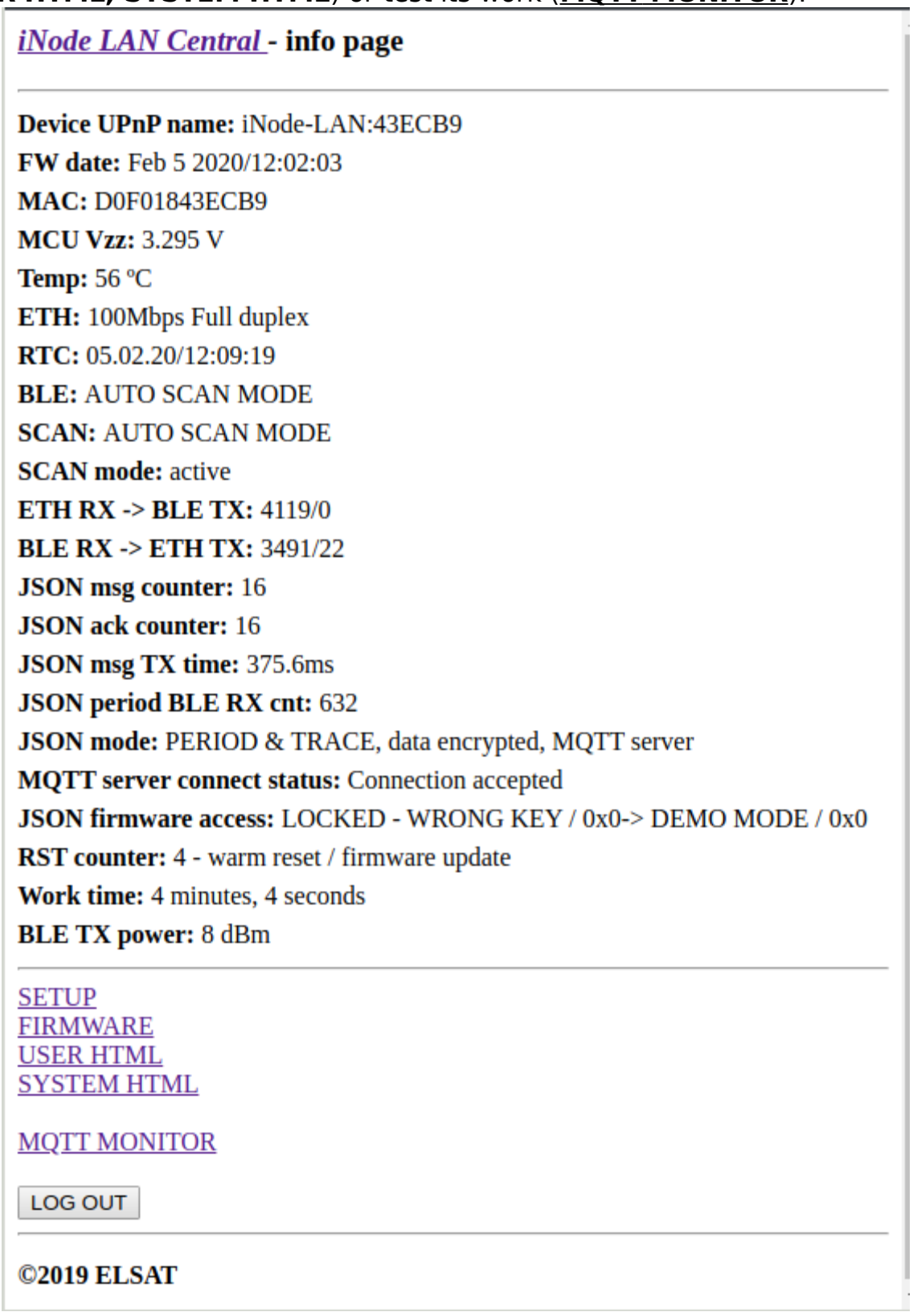
- These are primarily battery devices.
- Operate without replacing it for up to 12 months depending on the application and method of use.
- They have memory for recording events, measurement readings etc.
- Precise temperature, humidity, acceleration or magnetic field sensors allow for precise control of home automation or care for the elderly.
- As remote control devices, despite their low power consumption, they have a large range and features inaccessible to other competing devices - own user password, AES encryption, control directly from a smartphone.
- BT4.0 - LAN or BT4.0 - GSM gateways connect **iNode** sensors with the Internet.

iNode can also help control the movement of people or goods, saving the time of appearance and disappearance from the range of the recorder (active RFID® with a long range). New functionalities related to product development are also not a problem - it enables remote firmware exchange from a PC or smartphone with Bluetooth 4.0® and Bluetooth Low Energy® (Bluetooth Smart®) support.

iNode LAN Central enables the existence of devices with BLE (Bluetooth Smart, IoT - Internet of Things) in networks with the Ethernet protocol: LAN, Wi-Fi or Internet. **iNode LAN Central** will send to the MQTT or HTTP server data received from **iNode Care** sensors or other BLE tags. Data can be encrypted so that the user can use the public MQTT or HTTP server. Data decoding takes place only in the application, e.g. **iNode MQTT Monitor**.

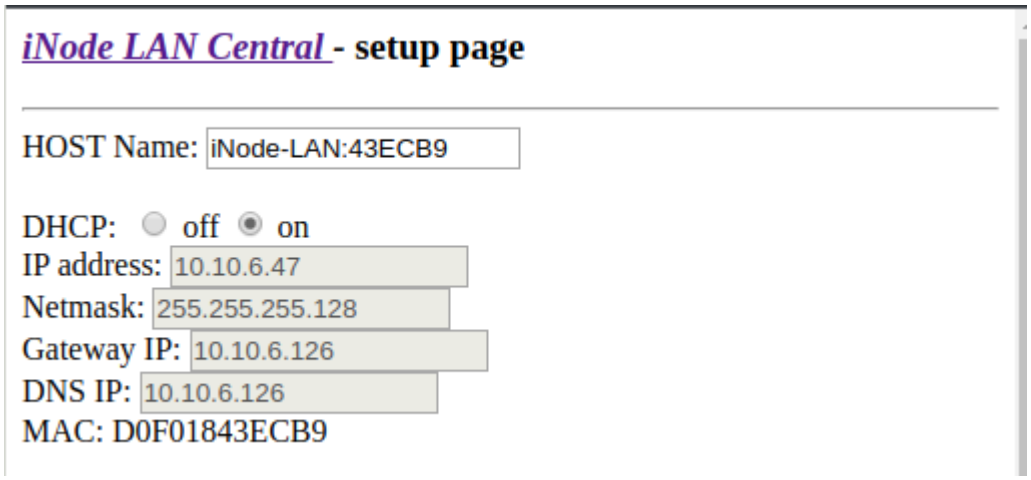
2. iNode LAN Central setup

The device has DHCP enabled by default - in this way it obtains the 10 / 100Mbps LAN address. After entering this IP address in the browser should display the following page, which displays statistical information about the device, its name, temperature, working time since the last reset, etc. You can select from it further pages for configuring the device (**SETUP**, **FIRMWARE**, **USER HTML, SYSTEM HTML**) or test its work (**MQTT MONITOR**).



2.1. SETUP

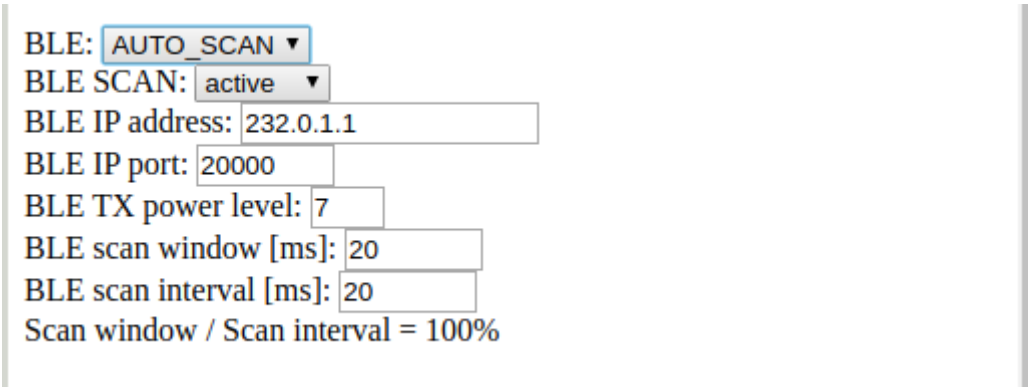
The **SETUP** page allows you to change the way the device obtains its IP address.



If ***DHCP off*** is selected, the fields ***IP address***, ***Netmask***, ***Gateway IP***, ***DNS IP*** are active and the addresses and values should be entered into them so that **iNode LAN** can work in the Ethernet network. By default, DHCP is enabled, which means that all these network parameters will be downloaded from the DHCP server, which is usually provided by e.g. an ADSL router.

The user can change the default device name in the ***HOST name*** field to any other name. It can be up to 16 characters long. It is inadvisable, for example, to use different than English diacritical marks because they can be misinterpreted by other network devices.

When it comes to BLE, **iNode LAN Central** can work in two modes:



1. **AUTO SCAN** - after powering, BLE environment is scanned in active mode and results are sent to LAN using IP/UDP as multicast/unicast or broadcast frames. You can easily receive them on any IP enabled device with any OS like Linux, Windows or Android. A data structure is something different from this given in **iNode Serial Transceiver USB/UART**. This mode is switched off when other client is connected to **iNode LAN Central** through TCP/IP socket.

IP multicast is a technique for one-to-many communication over an IP infrastructure in a network – defined by multicast group and port. For iNode-LAN it is 232.0.1.1:20000. Unicast messaging is used for all network processes in which a private or unique resource is requested. In computer networking, broadcasting refers to transmitting a packet that will be received by every device on the network. In practice, the scope of the broadcast is limited to a broadcast domain. Broadcast a message is in contrast to unicast addressing in which a host sends datagrams to another single host identified by a unique IP address.

2. **OFF** - after powering, device is not active in any manner in BLE environment but still can work with remote application. From a telnet program, e.g. Hyperterminal.exe, you can connect to **iNode LAN Central** on port 5500. It works the same as **iNode Serial Transceiver USB** via COM, e.g. it supports the same protocol.

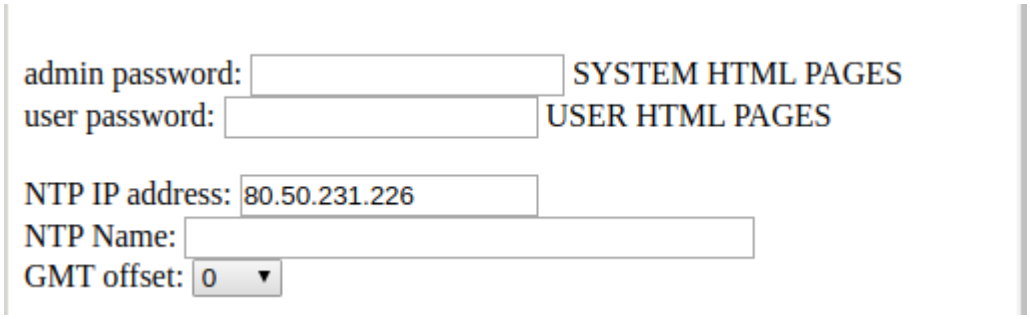
BLE Scan chose type of the BLE scanning – active or passive. In active mode **iNode LAN Central** sends request to each scanned device so you should remember that this type of scanning can drain more the battery of the scanned device.

BLE IP address and ***BLE IP port*** determine a receiving server IP address and port. **iNode LAN Central** will send to this server UDP datagrams with the received BLE data frames (in **AUTO SCAN** mode).

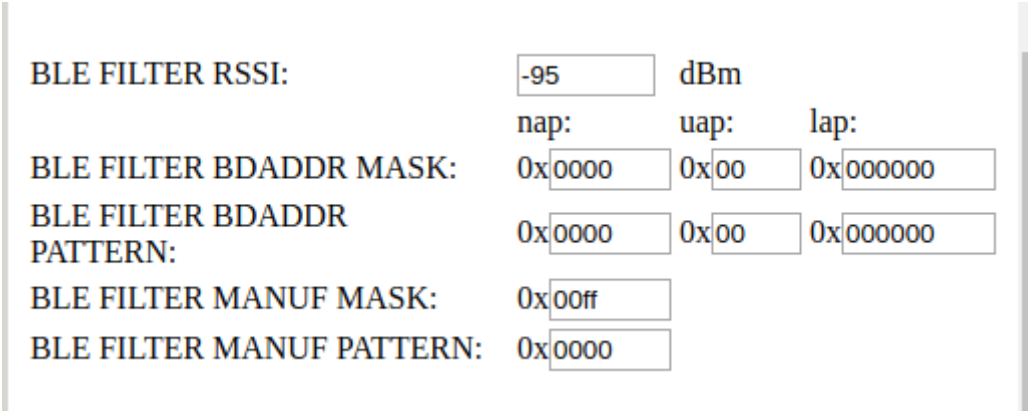
In ***BLE TX power level*** you can set the TX Power value during active scan. Relations between TX power level and TX Power in dBm is showed in a table below:

BLE TX power level	***TX Power [dBm]***
0	-18
1	-12
2	-10
3	-4
4	-2
5	+2
6	+6
7	+8

The values in the ***BLE scan window*** and ***BLE scan interval*** fields determine how long the device scans (***BLE scan window***).



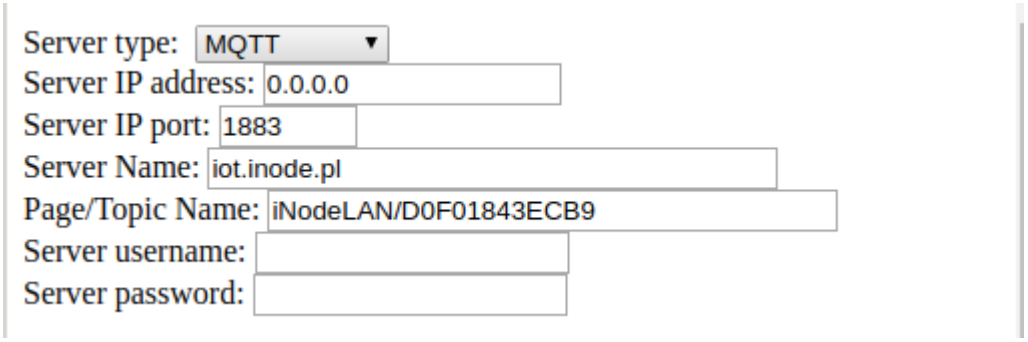
You can set the ***admin password*** (system HTML pages) and ***user password*** (user HTML pages) for built-in HTTP server.



The above parameters allow filtering BLE devices to send information about them in JSON format.
BLE FILTER RSSI – signal threshold level; further filters only take into account devices from which the received signal level is higher than that set here.
BLE FILTER BDADDR MASK – BDADDR address mask.
BLE FILTER BDADDR PATTERN – the pattern of the BDADDR address with which the received BDADDR is compared after the AND operation with the BDADDR mask.
BLE FILTER MANUF MASK – mask for Manufacturer Specific Data.
BLE FILTER MANUF PATTERN – pattern for Manufacturer Specific Data.

The ***NTP IP address*** field is used to enter the IP address of the NTP server. If the server is not found due to an incorrect address, the time in the device will not be correct, but the device will work. ***GMT Offset*** determines the hourly offset from the GMT time (time zone) in the range from -12 to 12 hours.

Next, we can set the parameters of the MQTT or HTTP / POST server with which the device is to work:

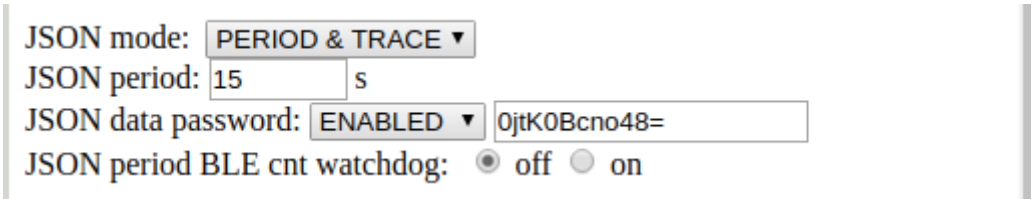


First of all, select the server type - ***Server type***, enter its IP address - ***Server IP address*** or name - ***Server Name***. Default device settings enable cooperation with the MQTT iNode server - iot.inode.pl

In the case of an HTTP server, enter its name - ***Server Name***, port (for HTTP should be port 80 by default) - ***Server IP*** port and the script name called by the device when sending data via POST - ***Page / Topic Name***.

Additionally, you can enter the username - ***Server username*** and password - ***Server password*** if access to the MQTT or HTTP / POST server requires it.

The following settings apply to the JSON data format:



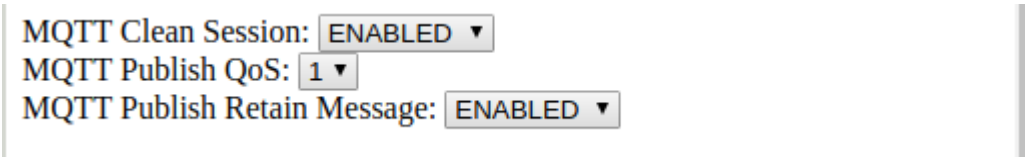
There are three possible JSON data collection modes – ***JSON mode***:

1. **STOP –** the device does not send data in JSON format.
2. **PERIOD & TRACE** – the device collects data about BLE devices in its environment and reaching it via LAN. In order for the device to be removed from the list, it must be out of range (not sending BLE packets for about 30 seconds). There can be approximately 24 devices on the list at the same time.
3. **PERIOD & CLEAR** - the device collects data about BLE devices in its vicinity and reaching it via LAN. After each sending of data in JSON format to the server, the list is deleted. There can be approximately 24 devices on the list at the same time.

The period of sending data is determined by entering the appropriate value expressed in seconds in the ***JSON period*** field. The minimum value is 2 seconds and depends on the speed of sending data to the MQTT or HTTP / POST server.

iNode LAN Central can encrypt the sent data in JSON format – ***JSON data password***. To do this, enable encryption – **ENABLED** and enter the password - maximum 16 characters. The same password must be entered into the **iNode MQTT Monitor** application later, so it can decode the data. During the operation of restoring the default settings of the device - power on with the button on the bottom of the device pressed - a new random password is created.

JSON period BLE cnt watchdog, if enabled, resets the device when no data is received from BLE or LAN during the ***JSON period***. Additionally, the device has a built-in functionality to reset the device after half of the lease time downloaded from DHCP.

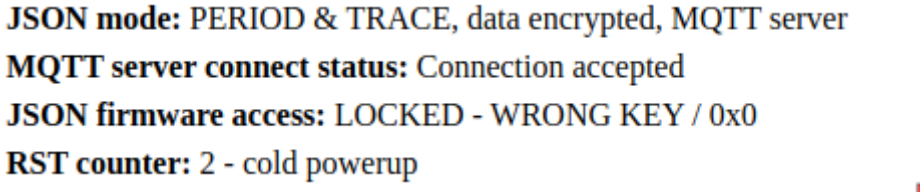


The above settings only apply to the MQTT server. If ***MQTT Publish Retain Message*** - is on - **ENABLED**, the last message sent is remembered by the MQTT server. The meaning of ***MQTT Publish QoS*** is as follows:

- ***QoS 0*** - the client will not receive any confirmation from the server. Similarly, the message delivered to the client from the server does not have to be confirmed. This is the fastest way to post and receive messages, but also the one where you will most likely lose messages.
- ***QoS 1*** - the client will receive a confirmation message from the server after it has been published. If the expected confirmation is not received within the specified time, the customer must retry the message. The message received by the client must also be confirmed in time, otherwise the server will deliver the message again.

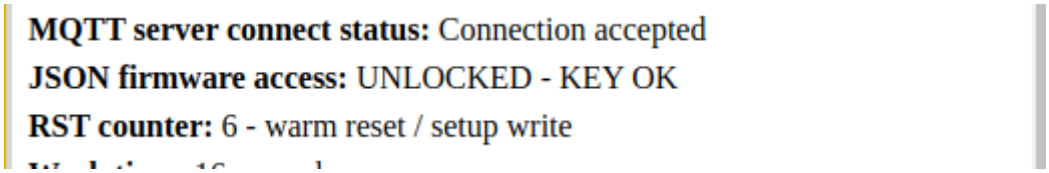
Firmware for **iNode LAN Central** can be uploaded to **iNode LAN**. However, if the **iNode LAN** does not contain the correct firmware unlock key - ***Firmware unlock key***, after 15 minutes from switching on, there will be no information about BLE devices in the data sent in JSON format:

JSON firmware access*: LOCKED - WRONG KEY / 0x0* – JSON does not contain BLE data; **iNode MQTT Monitor** does not show any BLE devices or shows a red crossed circle next to them.
JSON firmware access*: LOCKED - WRONG KEY / 0x0-> DEMO MODE / 0x0.* - JSON contains BLE data; 15 minutes after switching on the device.



Example of entering the code

and device status, if the entered code is correct:



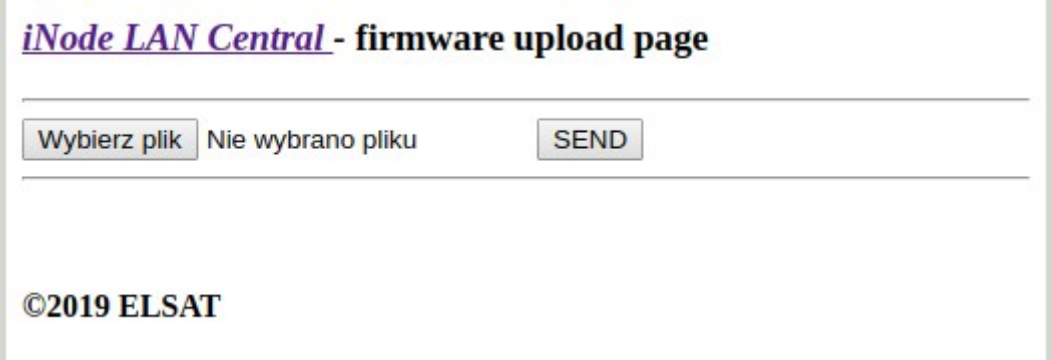
Each **iNode LAN Central** has factory unlocked firmware. You cannot enter a new key then. The key is not deleted even when replacing the firmware with the firmware for **iNode LAN**.

In order for the changed settings to be saved in the device, press the ***SAVE*** button. Correct input will be confirmed by the message ***done: OK***. After approximately 3-5 seconds, the device will reset for the new settings to take effect. When changing the parameters of the Ethernet network, be careful not to specify addresses outside the LAN.

The default settings can be restored by turning on the power of the device while pressing the RESET button located in the hole on the bottom of the device.

2.2. FIRMWARE

The **FIRMWARE** page allows changing the firmware in the device.

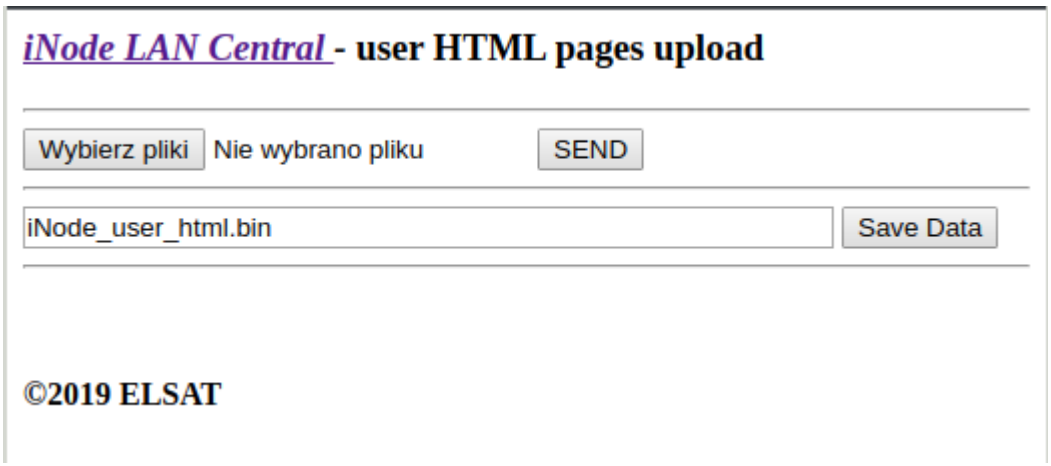


You can select a file with a new firmware after click on the ***Choose File*** button. Firmware uploading is done after click on the ***SEND*** button and a message ***uploading …*** is showing. After successful termination is replaced by ***done: OK*** and after a while by ***restarting …*** If firmware is correct (proper for this type of device) and operation will be successful a message ***done: OK*** will be showed. After about 3-5 seconds the device will be restarted and the new firmware will be used. In the case of enabled DHCP, wait for a while until the device receives the network parameters from the DHCP server again - the LED next to the RJ45 connector flashes quickly. Slow blinking of the LED indicates that the network parameters have been downloaded via DHCP.

Firmware ***fep*** files or applications can be downloaded from our technical support: http://support.inode.pl/ user ***inode***, without a password.

2.3. USER HTML

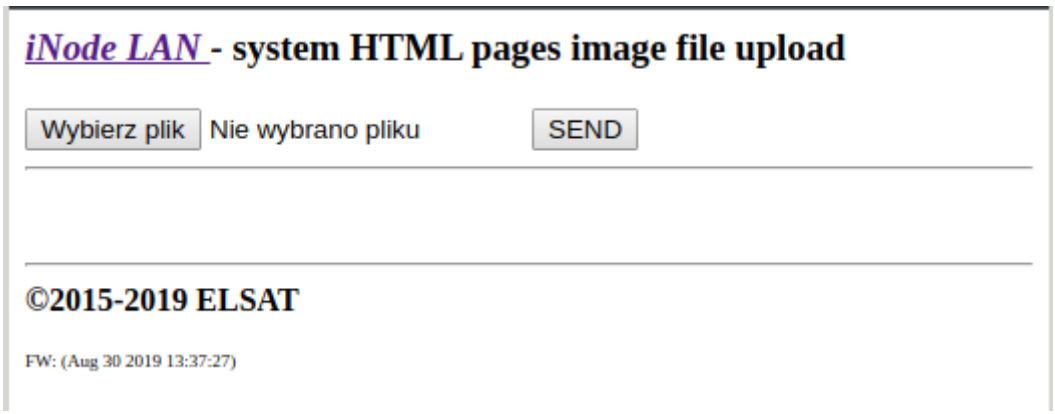
The **USER HTML** page allows you to enter your own user pages into the device. 2.9MB of memory is allocated for these pages. All files associated with the pages (images, scripts, etc.) should be placed in one directory. There can be a maximum of 512, and their names can be up to 40 characters.



You can select a HTML page files and other files (pictures, scripts etc.) or one binary image file after click on the ***Choose Files*** button. Uploading is done after click on the ***SEND*** button and a message ***reading files: done, uploading file of xxx kbytes*** is showing. After successful termination is replaced by ***done: OK.*** You can save on local disk drive binary image of the user HTML pages after click on the ***Save Data*** button.

2.4. SYSTEM HTML

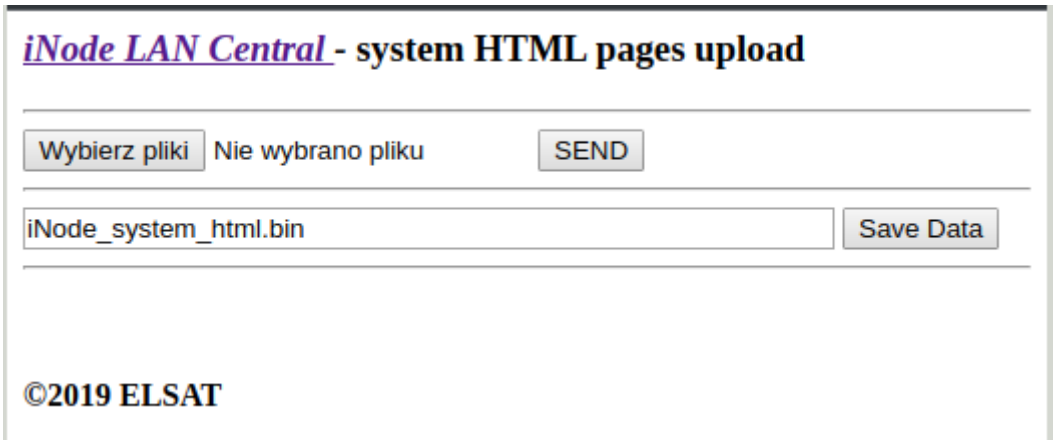
The **SYSTEM HTML** page allows you to exchange system pages. The file with pages in the same format as those of the user is uploaded to the system pages area using the flash.cgi page (it is always available directly). It can also be one file with bin pages.



You can select a HTML page files and other files (pictures, scripts etc.) or one binary image file after click on the ***Choose File*** button. Uploading is done after click on the ***SEND*** button and a message ***reading files: done, uploading file of xxx kbytes*** is showing. After successful termination is replaced by ***done: OK.*** At the bottom of page there is an information about creation date of the current firmware in a device: FW: (……)

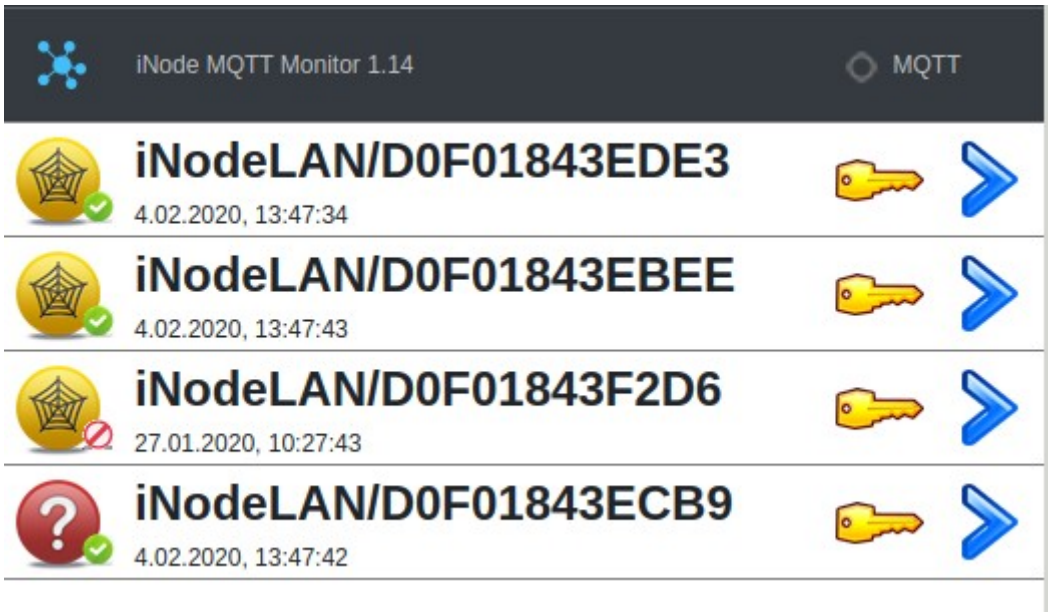
If there are no HTML system pages on your device, your browser may prompt you for a password or other errors. This usually happens when you upload a bin file with the user's HTML pages as system ones. Then, re-enter the system HTML pages into the device using *flash.cgi* page.

The user can create his own system pages image using the *system_flash_html.shtml* page. This is done in the same way as for the user's HTML pages.



2.5. MQTT MONITOR

The iNode MQTT MONITOR application allows you to test communication between the device and the MQTT or HTTP/POST server . The **iNode MQTT Monitor** application is dedicated to the Google Chrome browser and works on Android, Windows, Linux operating systems, etc. After loading the application, it can be installed for easier launch later. An application icon will appear on the main screen.

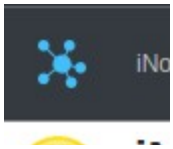


After starting the application, it shows devices that send data to the MQTT server iot.inode.pl. This is a free MQTT test server for users of the iNode products.

Important !

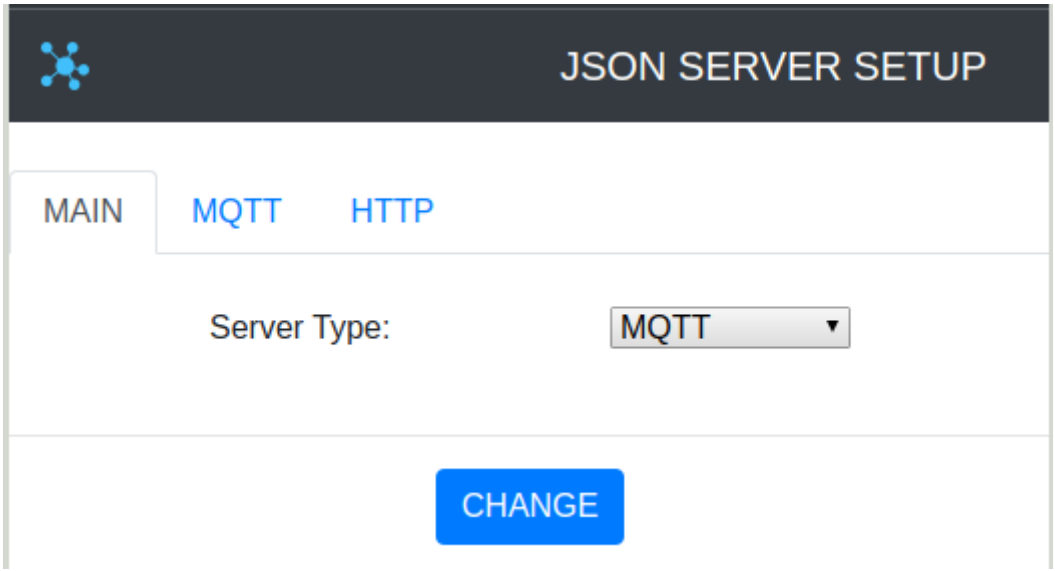
ELSAT s.c. does not guarantee that the MQTT iot.inode.pl server will be available in the future and under what conditions. The user must be aware that data sent to this server may be received by others. For privacy, you should ensure that send on this server data to be encrypted - this is the default option in the **iNode LAN Central** device. The default password for encrypting them is different and randomly created on each device. The data on the server are not archived in any way, but they are publicly available, which results from the specifics of the MQTT server operation if access to it is not restricted by means of a username and password. ELSAT s.c. is not responsible for the content of this data in any way and does not interfere in any way – moderates it.

Configuration of the **iNode MQTT Monitor** application is done by clicking on the image in the upper left corner of the screen:

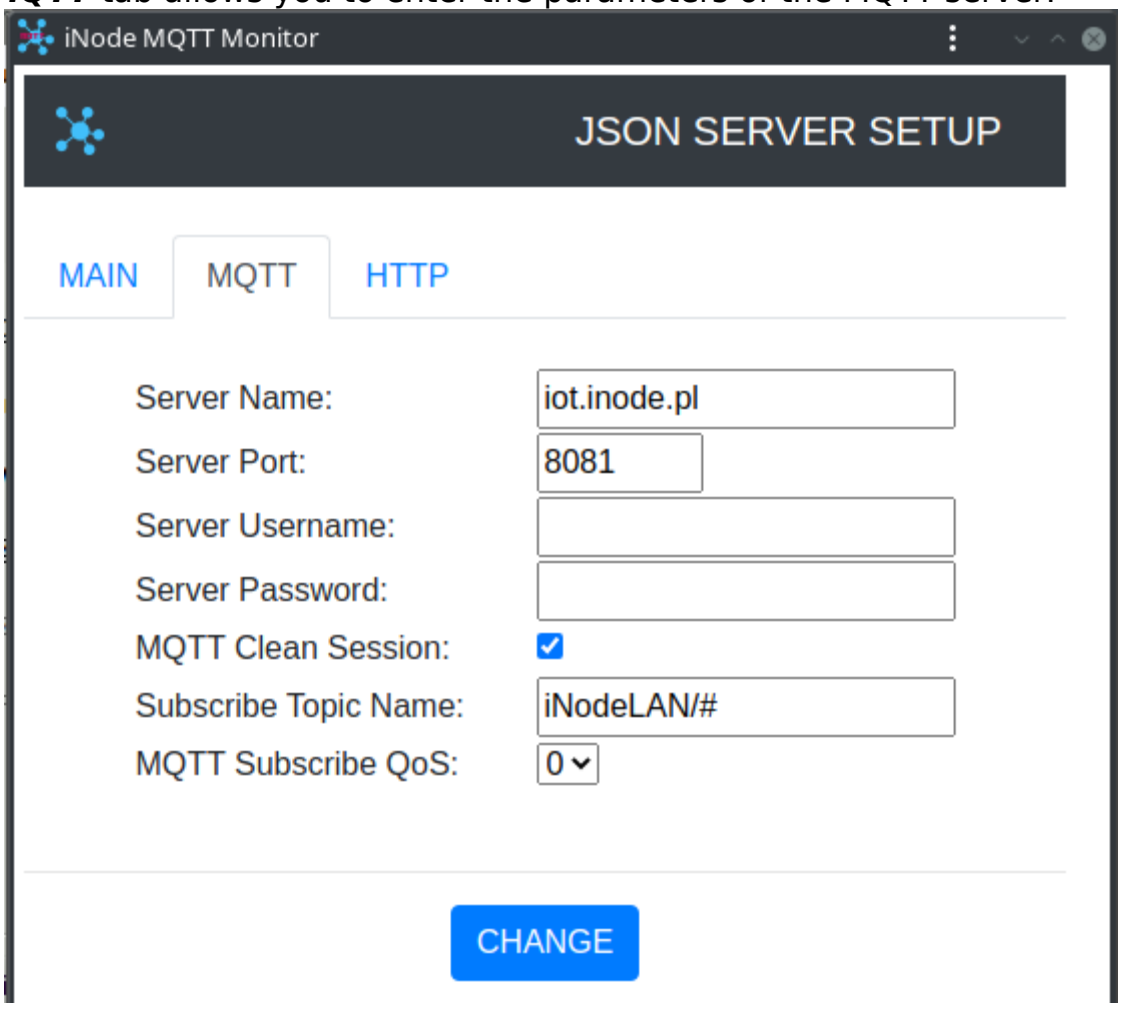


The following application screen will appear - **JSON SERVER SETUP**:

The ***MAIN*** tab allows you to select the type of server with which the application is to work. It can be HTTP, MQTT, USB or BLUETOOTH. The latter option is available from Google Chrome 79, however, so far it works only under Android. You may need to enable in **chrome://flags/#enable-experimental-web-platform-features** for USB or BLUETOOTH.

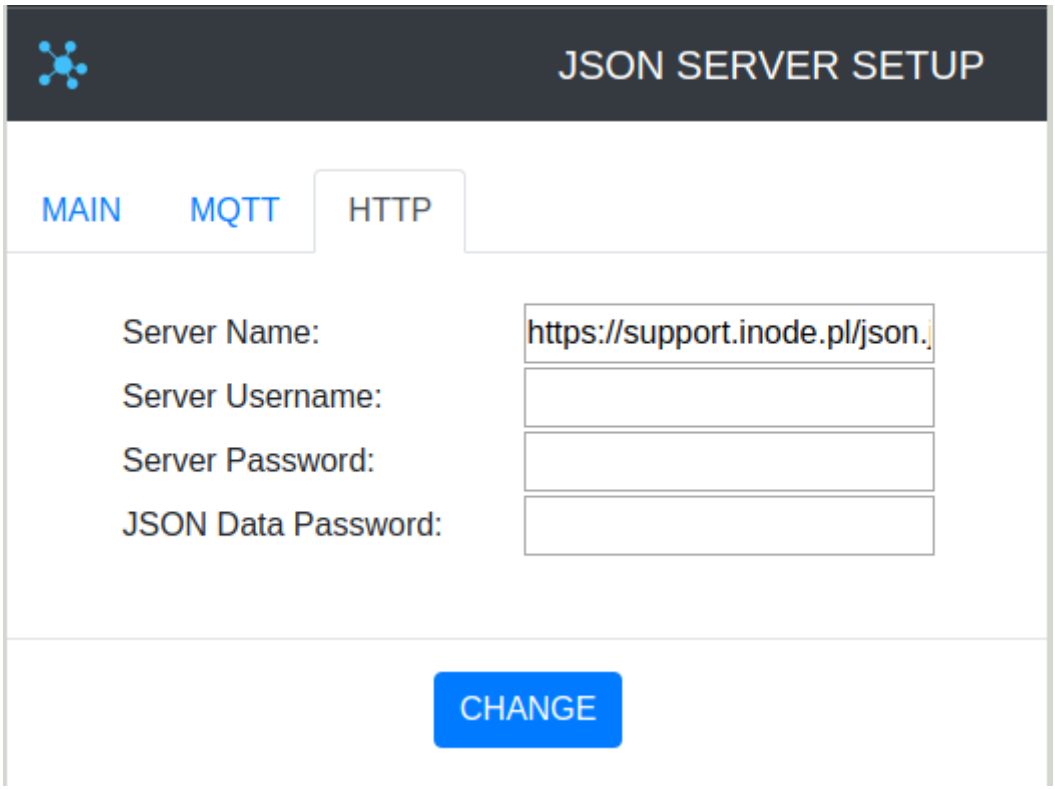


The ***MQTT*** tab allows you to enter the parameters of the MQTT server.



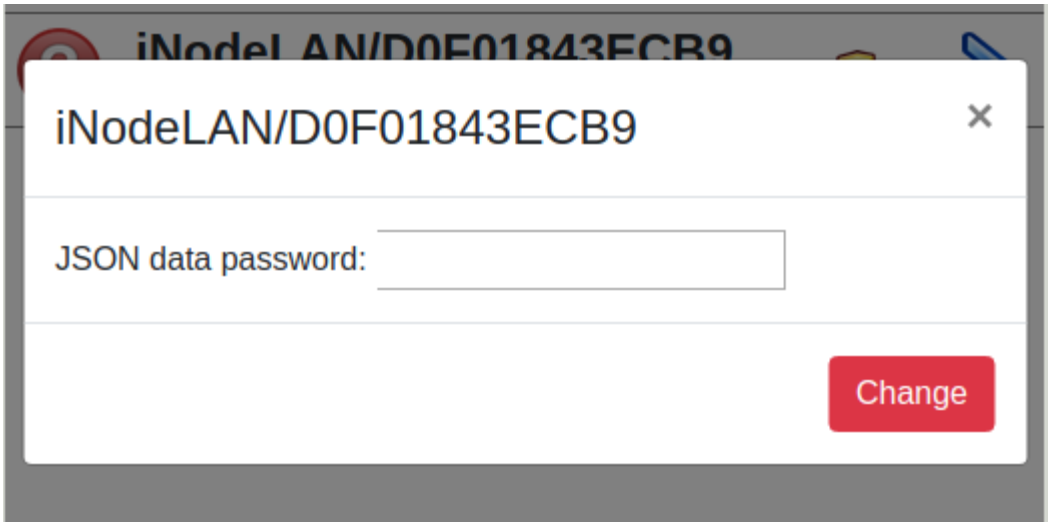
- ***Server Name*** – server name
- ***Server Port*** – the port at which the WebSocket service of the MQTT server is available
- ***Server Username*** – username if access to the MQTT server is restricted
- ***Server Password*** – password to access the MQTT server
- ***MQTT Clean Session*** - when the ***MQTT Clean Session*** flag is set, the client does not want a persistent session. If the client disconnects for any reason, all information and messages in the queue from the previous persistent session are lost.
- ***Subscribe Topic Name*** – it must be the same value as in the **iNode LoRa GSM MQTT** settings in the **PUBLISH – Topic** field or its fragment.
- ***MQTT Subscribe QoS:***
  - ***QoS 0*** - the client will not receive any confirmation from the server. Similarly, the message delivered to the client from the server does not have to be confirmed. This is the fastest way to post and receive messages, but also the one where you will most likely lose messages.
  - ***QoS 1*** - the client will receive a confirmation message from the server after it has been published. If the expected confirmation is not received within the specified time, the customer must retry the message. The message received by the client must also be confirmed in time, otherwise the server will deliver the message again.

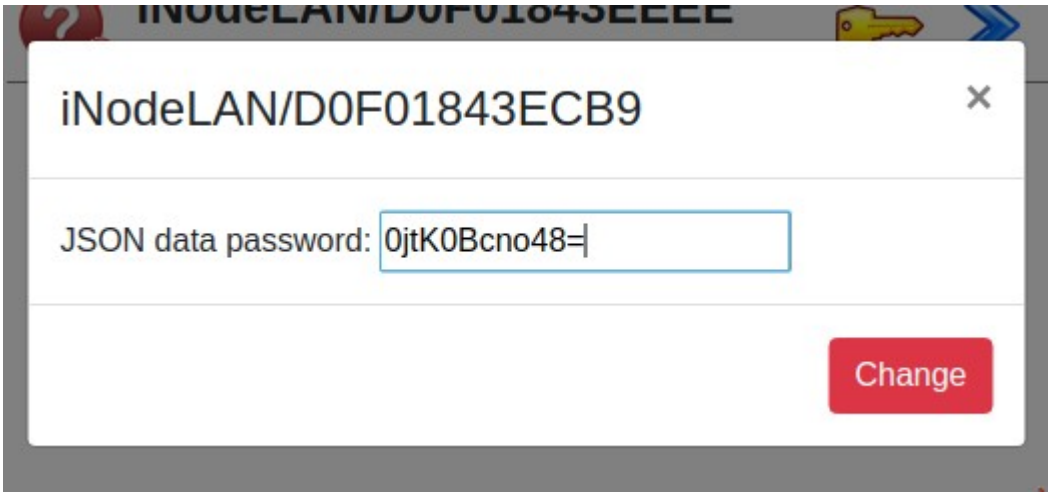
The ***HTTP*** tab allows you to specify HTTP server parameters.



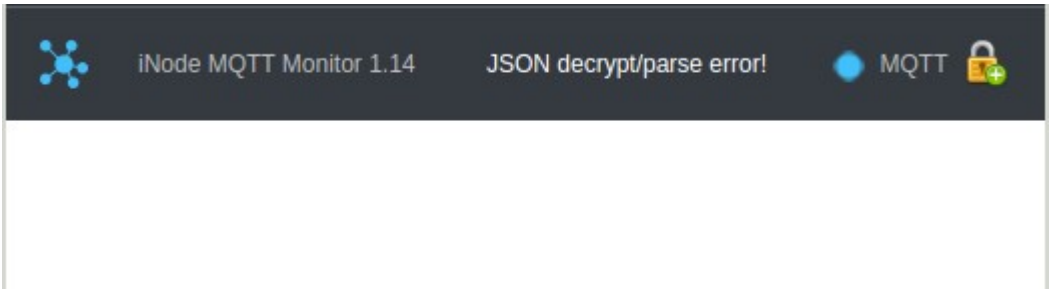
- ***Server Name*** – a full url containing the server name and path to the file with the given JSON
- ***Server Port*** – the port at which the HTTP server service is available
- ***Server Username*** – username if access to the HTTP server is restricted. Basic authorization type
- ***Server Password*** – password to access the HTTP server

If **iNode LoRa GSM MQTT** sends encrypted JSON data, then after selecting the picture with the key, enter the password to decode them and press the ***CHANGE*** button . It must be the same password as in the ***JSON*** tab in the ***KEY*** field.



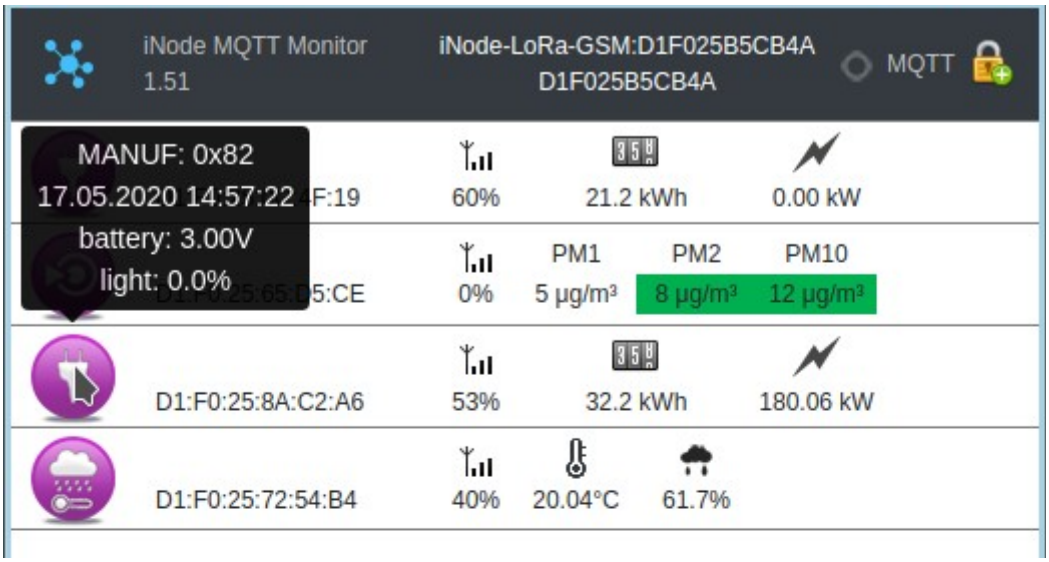


If the password is unset or wrong, after selecting the blue right arrow an error message will appear - ***JSON decrypt/parse error***. The fact that the data is encrypted is shown by a picture of the padlock in the upper right corner of the screen.



If the password was entered correctly, the application will display information about the devices from which they are sent by the given **iNode**

LoRa GSM MQTT. By hovering the mouse or touching individual elements (smartphone) additional information about a given device will be displayed.



3. JSON data format

3.1. Decrypted JSON data:

In the first position of the JSON data being sent - the ***data*** table, there is information about **iNode LAN Central**.

- timestamp – timestamp
- type – name
- mac – mac address
- rtc – device time in seconds
- ethRx – number of frames received on the LAN
- ethTx – number of frames sent over the LAN
- bleRx- number of frames received by BLE
- bleTx - the number of frames sent by BLE (only if active scanning is enabled)
- workTime – device operation time in seconds
- txp – set transmission power
- rst – number of device resets
- temp – device temperature in degrees Celsius
- msg – number of JSON data shipments
- ack – number of confirmed data shipments
- tx_time – JSON data sending time in microseconds
- juf – security information
- period – period of sending JSON data
- manuf – device type code
- rstr – reason for the last reset

{"data": [{"timestamp": "2020-02-05T13:10:35Z","type": "iNode-LAN:43ECB9","mac": "D0F01843ECB9","ip": "10.10.6.47","rtc": 1580908235,"ethRx": 85,"ethTx": 1,"bleRx": 114,"bleTx": 0,"workTime": 17,"txp": 8,"rst": 7,"temp": 56,"msg": 2,"ack": 1,"tx_time": 168508,"juf": 128,"period": 15,"manuf": 244,"rstr": 20},{"timestamp": "2020-02-05T13:10:34Z","mac": "D0F01843F3D8","rssi": -64,"rawData": "0201061107694E6F646520426561636F6E0000000003FF0080020A08","rawResp": "0D09694E6F64652D343346334438"},{"timestamp": "2020-02-05T13:10:35Z","mac": "35FD6890709F","rssi": -64,"rawData": "1EFF06000109200240FFC4543224734A4054BEABBFCF2DE413BBB209EC8750","rawResp": ""},{"timestamp": "2020-02-05T13:10:35Z","mac": "1DB6F43813AF","rssi": -66,"rawData": "1EFF060001092002468330 7B82213C7F54DD5BBB25CE60CE90ED7FC8E15B97","rawResp": ""},{"timestamp": "2020-02-05T13:10:34Z","mac": "FE2BD474F9EC","rssi": -57,"rawData": "031941030201060303E7FE09FFF8F8ECF974D42BFE0409503131","rawResp": ""},{"timestamp": "2020-02-05T13:10:35Z","mac": "D0F01843DB0F","rssi": -57,"rawData": "02010603FFC088020AFE0D09694E6F64652D343344423046","rawResp": ""},{"timestamp": "2020-02-05T13:10:34Z","mac": "D0F01843F3DA","rssi": -79,"rawData": "02010619FFC09B01B000000000931800000F00CCF6B00428FEE4821F30","rawResp": "0D09694E6F64652D343346334441020AFE"},{"timestamp": "2020-02-05T13:10:33Z","mac": "D0F01843F15C","rssi": -77,"rawData": "0201060EFFA0820000C31300006440B100A0020A08","rawResp": "0D09694E6F64652D343346313543"},{"timestamp": "2020-02-05T13:10:33Z","mac": "D8A01D6B8E1A","rssi": -79,"rawData": "0F0843656E7472616C2D364238453141","rawResp": ""},{"timestamp": "2020-02-05T13:10:35Z","mac": "30F7724CCFF0","rssi": -83,"rawData": "1AFF4C00021550765CB7D9EA4E2199A4FA879613A49270D8B708CE","rawResp": ""},{"timestamp": "2020-02-05T13:10:34Z","mac": "D0F01843F3DB","rssi": -84,"rawData": "02010619FF909B01C000000000471836100F0090F6D67324F4862721D3","rawResp": "0D09694E6F64652D343346334442020AFE"},{"timestamp": "2020-02-05T13:10:34Z","mac": "D0F01843F1E4","rssi": -52,"rawData": "0201060EFF908200004D000000E803B20080020AFE","rawResp": "0D09694E6F64652D343346314534"},{"timestamp": "2020-02-05T13:10:34Z","mac": "D0CF5E03930E","rssi": -66,"rawData": "0201060EFFA08200009F000000E803A00000020A14","rawResp": "0D09694E6F64652D303339333045"},{"timestamp": "2020-02-05T13:10:34Z","mac": "D0F01843E2C5","rssi": -74,"rawData": "02010619FF129D01B00004643F3519D812390034FE30FD8292C68368FC","rawResp": "0D09694E6F64652D343345324335020A06"},{"timestamp": "2020-02-05T13:10:34Z","mac": "D0F01843F2FA","rssi": -72,"rawData": "0201060EFFA0820000010000000E843B00000020AFE","rawResp": "0D09694E6F64652D343346324641"},{"timestamp": "2020-02-05T13:10:34Z","mac": "D0F01843F3D5","rssi": -74,"rawData": "0201061107694E6F646520426561636F6E0000000003FF0080020AFE","rawResp": "0D09694E6F64652D343346334435"},{"timestamp": "2020-02-05T13:10:34Z","mac": "D0F01843F2D9","rssi": -70,"rawData": "02010619FF1A9500C00000B807F81F46001B001ADC26436A4EC8AAD4CF","rawResp": "0D09694E6F64652D343346324439020A08"},{"timestamp": "2020-02-05T13:10:34Z","mac": "D0CF5E039918","rssi":

-74,"rawData": "02010619FF009BFF80000000009E1813130000A9FEC617BB7D98BD395D","rawResp": "0D09694E6F64652D303339393138"},{"timestamp": "2020-02-05T13:10:33Z","mac": "D0F01843F3DC","rssi": -79,"rawData": "02010619FF909101B00000077CA08000000F0090F605A530C80FB0D0F4","rawResp": "0D09694E6F64652D343346334443020AFE"},{"timestamp": "2020-02-05T13:10:35Z","mac": "D0F01843F150","rssi": -79,"rawData": "02010619FF149E00C000006D02A10000000900A24846D5481621C47BE9","rawResp": "0D09694E6F64652D343346313530020A08"},{"timestamp": "2020-02-05T13:10:33Z","mac": "D0F01843F458","rssi": -79,"rawData": "02010619FF129D01C000047C3FD818C80E0000163E538DEF07F9AF5D84","rawResp": "0D09694E6F64652D343346343538020AFE"},{"timestamp": "2020-02-05T13:10:34Z","mac": "0CF3EEA355A1","rssi": -74,"rawData": "02010417FF0C0300B8DC2023FEE4110FBE2000145E45041DBF000003030118","rawResp": ""}]}

3.2. Processed JSON data

3.3. Encrypted JSON data:

If the data from **iNode LAN Central** is encoded at the beginning of the JSON file, there is a ***key*** field. This is a temporary key used to encrypt JSON data. It is encrypted with the master key entered into **iNode LAN Central**. The data is encrypted with the ARCFour algorithm.

{"key": "33FE46D546832247CC91A8EA733D56E9","data": [Q}H/k_ {.m Z u Ö-
G z|Tw e Zn&v1 T} qE l m qq O J)};)x m n
ت+e UC IA7!j]+/k'v 0Ee3@^f[+">
y aK B\ 9:l Bٿ]k A6 Y(}w |SL_h)
j .xOLT S@
T h _O
Q1]}

3.4. JSON data decryption

Below is an example function in JavaScript that decodes encrypted JSON data. The jsaes.js file can be downloaded from:
https://support.inode.pl/apps/iNodeMqttMonitor/js/jsaes.js

```
/*
 * RC4 symmetric cipher encryption/decryption
 * @license Public Domain
 * @param string key - secret key for encryption/decryption
 * @param string str - string to be encrypted/decrypted
 * @return string
 */

var rc4_s = [];
var rc4_i;
var rc4_j;

function rc4_init(rc4_key, rc4_key_length)
{
   var j = 0, x;
   for (var i = 0; i < 256; i++) {
      rc4_s[i] = i;
   }
   for (i = 0; i < 256; i++) {
      j = (j + rc4_s[i] + rc4_key[i % rc4_key_length]) % 256;
      x = rc4_s[i];
      rc4_s[i] = rc4_s[j];
      rc4_s[j] = x;
   }

   rc4_i=0;
   rc4_j=0;
}

function rc4_get_xor_byte()
{
   var x;

   rc4_i = (rc4_i + 1) % 256;
   rc4_j = (rc4_j + rc4_s[rc4_i]) % 256;
   x = rc4_s[rc4_i];
   rc4_s[rc4_i] = rc4_s[rc4_j];
   rc4_s[rc4_j] = x;
   return rc4_s[(rc4_s[rc4_i] + rc4_s[rc4_j]) % 256];
}

var JSON_USER_KEY = new Array(16);
var JSON_DECRYPT_KEY = new Array(16);

function searchCommentValue(sstr, key)
{
   var offset_start=sstr.search(key);

   if(offset_start>=0)
   {
      var rsstr=sstr.slice(offset_start+key.length);
      var offset_end=rsstr.search('"');
      return rsstr.substring(0,offset_end);
   }
   else
```

```
    {
        return "";
    }
}

function decodeJSON(json_raw, json_key)
{
    var img_dataView = new DataView(json_raw);
    var ik;
    var img_byte;
    var xor_byte=0;
    var ik_img_offset=0;

    for(var i = 0; i < 16; i++)
    {
        JSON_USER_KEY[i] = 0;
    }

    for(var i = 0; i < json_key.length; i+=2)
    {
        JSON_USER_KEY[15-i] = json_key.charCodeAt(i+1);
        JSON_USER_KEY[14-i] = json_key.charCodeAt(i);
    }

    var JSON_KEY=searchCommentValue(ab2str(json_raw),'{"key": "');

    if(JSON_KEY.length!=0)
    {
        AES_Init();

        var block = new Array(16);
        for(var i = 0; i < 16; i++)
          { block[15-i] = parseInt(JSON_KEY.substr(i*2,2), 16) };

        var key = new Array(16);
        for(var i = 0; i < 16; i++)
            { key[i] = JSON_USER_KEY[i]; }

        AES_ExpandKey(key);
        AES_Decrypt(block, key);

        for(var i = 0; i < 16; i++)
            { JSON_DECRYPT_KEY[15-i] = block[i] };

        AES_Done();

        rc4_init(JSON_DECRYPT_KEY,16);

        var json_data_start=ab2str(json_raw).search(',"data": ')+10;

         for(ik=json_data_start;ik<(img_dataView.byteLength-2);ik++)
        {
            img_byte=img_dataView.getUint8(ik);

            img_dataView.setUint8(ik,(img_byte^rc4_get_xor_byte())& 0xff);
        }
    }
    return json_raw;
}
```

3.5. BLE data decoding:

Data coding scheme in advertisement frame and response for active scan. Information about **AD Type** codes can be found in a Core_V4.0.pdf: Volume 3 Part C, Section 8. and at the page https://www.bluetooth.org/en-us/specification/assigned-numbers/generic-access-profile

advertisement frame:

02010619FF1293011000001700AB18951F485435BE5B809D6F571E40E8FE0000

020106

02 -> length of the data field: 2 bytes

0106 -> data

01 -> 0x01 -> EIR Data Type = 0x01 -> «Flags»

06 -> 0x06 -> EIR Data = 0x06 -> LE General Discoverable Mode (bit 1), BR/EDR Not Supported (bit 2)

19FF1293011000001700AB18951F485435BE5B809D6F571E40E8

19 -> length of the data field: 25 bytes

FF1293011000001700AB18951F485435BE5B809D6F571E40E8 -> data (25 bytes)

FF -> 0xFF -> EIR Data Type = 0xFF «Manufacturer Specific Data»

1293011000001700AB18951F485435BE5B809D6F571E40E8->

1293 -> 0x9312 -> 0x93XX iNodeCareSensor #3 identifier; 0xXX1X version 1; 0xXXX2 since last memory readout lasts 24 hours;

0110 -> 0x1001 type -> bit 15 to bit 12 -> reserved, bit 11 to bit 0 -> sensor group address

0000 -> 0x0000 flags ->

SENSOR_ALARM_MOVE_ACCELEROMETER=1,
SENSOR_ALARM_LEVEL_ACCELEROMETER=2,
SENSOR_ALARM_LEVEL_TEMPERATURE=4,
SENSOR_ALARM_LEVEL_HUMIDITY=8,
SENSOR_ALARM_CONTACT_CHANGE=16,
SENSOR_ALARM_MOVE_STOPPED=32,
SENSOR_ALARM_MOVE_GTIMER=64,
SENSOR_ALARM_LEVEL_ACCELEROMETER_CHANGED=128,
SENSOR_ALARM_LEVEL_MAGNET_CHANGE=256,
SENSOR_ALARM_LEVEL_MAGNET_TIMER=512

1700 -> 0x0017 value1

/* motion sensor */
0x8000 sensor is in move (bit 15 =1)
bit 14 to 10:
X-axis (5 bit value as 2's complement number) -> 0x00= 0
bit 9 to 5:
Y-axis (5 bit value as 2's complement number) -> 0x00= 0
bit 4 to 0:
Z-axis (5 bit value as 2's complement number) -> 0x17= -9

AB18 -> 0x18AB value2

/* temperature sensor */
Temperature= ((175.72 * Temp_Code)/65536)-46.85 [°C]
Temp_Code = 0x18AB *4 = 0x62AC = 25260
Temperature = 20,879 °C

951F -> 0x1F95 value3

/* humidity sensor */
%RH= ((125*RH_Code)/65536)-6 [%]

RH_Code = 0x1f95 *4 = 0x7e54 = 32340
%RH= 55,68 %

485435BE -> 0x5448BE35 time (time stamp; number of seconds since 01.01.1970)

5B80 9D6F 571E 40E8 -> an AES128 digital signature of all data

response for an active scan:

0D09694E6F64652D333536313441020A020000000000000000000000000000000

0D09694E6F64652D333536313441

0D -> length of the data field: 13 bytes

09694E6F64652D333536313441 -> data

09 -> 0x09 -> EIR Data Type = 0x09 -> «Complete Local Name»

694E6F64652D333536313441 -> iNode-35614A

020A02

02 -> length of the data field: 2 bytes

0A02 -> data

0A-> 0x0A -> EIR Data Type = 0x0A -> «Tx Power Level»

02 -> 0x02 -> Tx Power Level = +2dBm

4. Technical info

Radio parameters:

- RX/TX:
  - BLE: 2402-2480 MHz
- output power (maximum):
  - BLE: +8dBm
- modulation:
  - BLE: GFSK
- antenna:
  - PCB internal MIFA type, 1,6dBi (POE version)
  - SMA external RP SMA MALE type, 3dBi (POE SMA version)

Software options:

- configurable from your PC:
  - BLE mode: AUTOSCAN
  - JSON work mode: MQTT lub HTTP
  - the BLE power with which the device operates from -18dBm to + 8dBm (maximum range up to 200 m in open space)
  - LAN settings: IP address (static or dynamic - DHCP), network mask, gateway, DNS, NTP server;
  - the name of the device in LAN & BLE
  - IP address & port for MQTT server connection
  - user password
  - admin password

Power supply:

- **POE** or **POE SMA** version:
  - active POE in the IEEE 802.3af 48V DC 1W standard;

Housing:

- metal case;
- dimensions: 81 mm x 38 mm x 22 mm;

Other:

- ratio scan window/scan interval = 1 -> receiving from BLE all the time;
- remote firmware update using web browser;
- remote control using TCP/IP telnet connection at port 5500 (the same control features like using WebSocket);
- LEDs: Ethernet LINK i STATUS;
- HTTP server:
  - 2,9MB for user HTTP pages (www) and 1MB for HTTP system pages (www);
  - max. 2 sockets simultaneously;
- RJ-45 connector 10Mbps/100Mbps Ethernet, 10BaseT; protocols: ARP, SSDP, UDP, TCP/IP, DHCP, SNTP, HTTP, MQTT;
- reset button (restore factory settings);
- temperature sensor with a resolution of 1 °C;
- operating temperature: from -20 to 45 °C;
- humidity: 35-80 % RHG.
- weight: 45 g;

Equipment:

- SMA antenna type RP SMA MALE, 3dBi (POE SMA version);

Software:

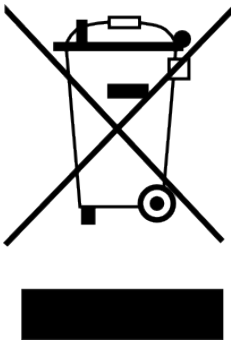
- any web browser;

Chipset:

- CSR1010;
- W5500;

The manufacturer reserves the right to change the parameters of the device and software, and to introduce other design solutions.

5. CORRECT PRODUCT REMOVAL (WASTE ELECTRICAL AND ELECTRONIC EQUIPMENT)



The packaging materials are 100% suitable for use as a secondary raw material. The packaging should be disposed of in accordance with local regulations. Keep packaging materials out of the reach of children as they pose a source of danger. The marking on the product or in related texts indicates that the product should not be disposed of with other household waste after it has expired. To avoid harmful effects on the environment and human health due to uncontrolled waste disposal, please separate the product from other types of waste and recycle responsibly to promote the reuse of material resources as a permanent practice.

Correct disposal of the device:

- Pursuant to the WEEE Directive 2012/19 / EU, the symbol of the crossed wheeled waste container means all electrical and electronic devices subject to selective collection.
- After the end of its useful life, this product must not be disposed of as normal household waste, but should be sent to a collection point for the recycling of electrical and electronic equipment. This is indicated by the symbol of the crossed-out wheeled waste container, placed on the product or in the operating instructions or packaging.
- The materials used in the device are reusable according to their designation. Thanks to the reuse, use of materials or other forms of use of used devices, you make a significant contribution to the protection of our natural environment.
- For information on the appropriate disposal point for used electrical and electronic equipment, please contact your local municipality administration or the device seller.
- Used, fully discharged batteries and accumulators must be disposed of in specially marked containers, taken to special waste collection points or sellers of electrical equipment.
- Users in companies should contact their supplier and check the terms of the purchase contract. The product should not be disposed of with other household waste.

Numer Deklaracji 2/10/2019 Data wystawienia Deklaracji 10.10.2019 r.
Number of declaration of Conformity *Date of issue of declaration*

DEKLARACJA ZGODNOŚCI WE
EC DECLARATION OF CONFORMITY

My/*We*: **ELSAT s.c.**
(nazwa producenta / producer's name)
ul.Warszawska 32E/1, 05-500 Piaseczno k/Warszawy
(adres producenta / producent's address)

niniejszym deklarujemy, że następujący wyrób:
declare, under our responsibility, that the electrical product:

iNode LAN Central — 0x0C00
POE; POE SMA;
(nazwa wyrobu / product's name) — *(model / model)*

spełnia wymagania następujących norm zharmonizowanych:
to which this declaration relates is in conformity with the following harmonized norm:

Radio Spectrum ISM (Article 3.2 of the RED directive):
PN-ETSI EN 300 328 V2.1.1:2016-11
EMC (Article 3.1.b of the RED directive):
PN-ETSI EN 301 489-1 V2.2.0:2017-03
PN-ETSI EN 301 489-17 V3.2.0:2017-03
Safety (Article 3.1.a of the RED directive):
PN-EN 62368-1:2015-03
Health (Article 3.1.a of the RED directive):
PN-EN 62311:2008
RoHs:
PN-EN IEC 63000:2019-01

jest zgodny z postanowieniami następujących dyrektyw Unii Europejskiej:
is compatible with the following European Union directives:

Dyrektywa RED 2014/53/UE
Dyrektywa EMC 2014/30/UE
Dyrektywa LVD 2014/35/UE
Dyrektywa RoHS 2011/65/UE

Procedura oceny zgodności: wewnętrzna kontrola produkcji zgodnie z załącznikiem II RED
Acceptance procedure: internal production control in accordance with Annex II of the RED Directive

10.10.2019 r. Paweł Rzepecki

Piaseczno k/Warszawy
(data i miejscowość / date and place)

Współwłaściciel
(podpis i stanowisko / signature and function)

RoHS 2 2011/65/EU