

Integracja z protokołem ZigBee

Numer dokumentu: PO-197 Wersja: 4.0 Data publikacji: 25 września 2025

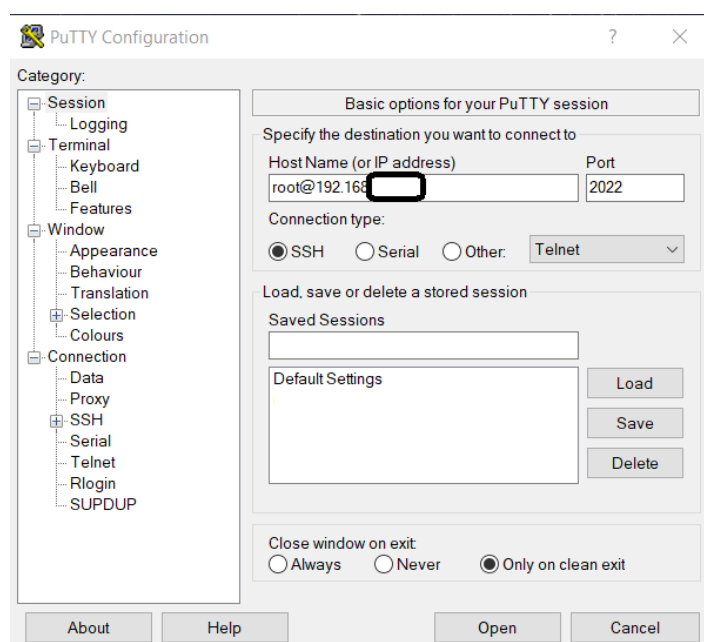
UWAGA: Z dniem 25 września 2025 r. wsparcie techniczne dla integracji z protokołem Zigbee zostaje zakończone. Po tej dacie, wszelkie problemy wynikające z jej stosowania nie będą objęte obsługą techniczną.

Wstęp

Integracja urządzeń wspierających protokół ZigBee z systemem Ampio możliwa jest na przykład dzięki dopięciu bramki do modułu M-SERV-s. W celu wykonania połączenia niezbędne jest użycie platformy Node-RED. W poniższym przykładzie jako bramka służy moduł ZBDongle-E firmy Sonoff z dedykowaną anteną.

Podłączenie bramki

Aby podłączyć bramkę, odłączamy M-SERV od zasilania, wpinamy moduł w dowolne złącze USB i zasilamy ponownie serwer. Po kilku minutach, poprzez interfejs [www](#) aktywujemy połączenie SSH (wskazówki dostępne w poradniku [Konfiguracja serwera](#)). Logujemy się przy użyciu utworzonego hasła na konto *root* na serwerze np. poprzez aplikację *putty*.



Wyszukanie portu z bramką

Po wpisaniu hasła wyszukujemy urządzenia z użyciem komendy:

```
dmesg | grep tty
```

Mostek prawdopodobnie zostanie dodany jako *ttyACM0*.

Konfiguracja dla obrazów serwera od wersji numer 400

Instalacja

Wchodzimy w tryb zapisu i odczytu

```
/opt/ampio/bin/rw
```

przechodzimy do folderu, w którym możemy wprowadzać zmiany

```
cd /root
```

aktualizujemy listę dostępnych pakietów

```
sudo apt-get update
```

instalujemy pakiet pnpm

```
npm install -g pnpm@10.4.1
```

instalujemy git

```
sudo apt install git
```

tworzymy folder

```
sudo mkdir /root/zigbee2mqtt
```

zmieniamy uprawnienia do folderu

```
sudo chown -R root:root /root/zigbee2mqtt
```

klonujemy repozytorium zigbee2mqtt

```
git clone --depth 1 https://github.com/Koenkk/zigbee2mqtt.git /root/zigbee2mqtt
```

przechodzimy do folderu

```
cd /root/zigbee2mqtt
```

instalujemy dedykowane zależności

```
pnpm i --frozen-lockfile
```

uruchamiamy budowanie paczki

```
pnpm run build
```

kopiujemy zawartość przykładu do naszego pliku konfiguracyjnego

```
cp /root/zigbee2mqtt/data/configuration.example.yaml /root/zigbee2mqtt/data/configuration.yaml
```

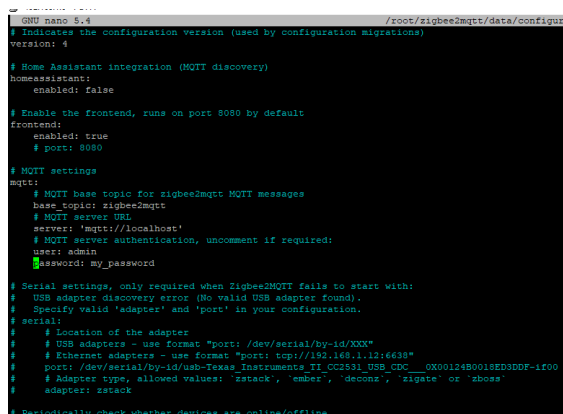
Modyfikacja pliku konfiguracyjnego

Otwieramy plik do edycji

```
nano /root/zigbee2mqtt/data/configuration.yaml
```

Pole `server` ustawiamy na `mqtt://localhost`.

Pola połączenia MQTT zgodnie z naszymi ustawieniami serwera, `user` to najczęściej `admin` oraz odpowiednie hasło (dane logowania jak do blochków `mqtt` w Node-RED). Należy pamiętać, aby usunąć znak `#` oznaczający komentarz linii.

A screenshot of a terminal window showing the configuration.yaml file being edited with nano. The file contains settings for Zigbee2MQTT, including MQTT base topic, server URL, MQTT server authentication, and serial settings. The user has uncommented the MQTT server authentication fields (user: admin, password: my_password).

```
# Indicates the configuration version (used by configuration migrations)
version: 4

# Home Assistant integration (MQTT discovery)
homeassistant:
  enabled: false

# Enable the frontend, runs on port 8080 by default
frontend:
  enabled: true
  # port: 8080

# MQTT settings
mqtt:
  # MQTT base topic for zigbee2mqtt MQTT messages
  base_topic: zigbee2mqtt
  # MQTT server URL
  server: 'mqtt://localhost'
  # MQTT server authentication, uncomment if required:
  user: admin
  password: my_password

# Serial settings, only required when Zigbee2MQTT fails to start with:
# USB adapter discovery error (No valid USB adapter found).
# Specify valid 'adapter' and 'port' in your configuration.
# serial:
#   # Location of the adapter
#   # USB adapters - use format "port: /dev/serial/by-id/XXXX"
#   # Ethernet adapters - use format "port: tcp://192.168.1.12:6638"
#   port: /dev/serial/by-id/usb-Texas_Instruments_TI_CC2531_USB_CDC___0X00124B0018ED3DDF-1F00
#   # Adapter type, allowed values: 'zstack', 'ember', 'deconz', 'zigate' or 'zboas'
#   adapter: zstack

# Periodically check whether devices are online/offline
```

Po zmianie zapisujemy i zamykamy plik konfiguracyjny. W putty robimy to poprzez `Ctrl+x`, następnie `y` i `Enter`.

Pierwsze uruchomienie

Wpisujemy komendę

```
pnpm start
```

Konfiguracja dla obrazów starszych niż 400

Tworzenie i konfigurowanie folderu

Tworzymy folder

```
sudo mkdir /ampio/rw/zigbee2mqtt
```

Nadajemy uprawnienia

```
sudo chown -R ${USER}: /ampio/rw/zigbee2mqtt
```

Klonowanie repozytorium zigbee2mqtt

```
git clone --depth 1 https://github.com/Koenkk/zigbee2mqtt.git /ampio/rw/zigbee2mqtt
```

Instalowanie zawartości

Zmieniamy aktualny folder

```
cd /ampio/rw/zigbee2mqtt
```

Następnie instalujemy

```
npm ci
```

Modyfikacja pliku konfiguracyjnego

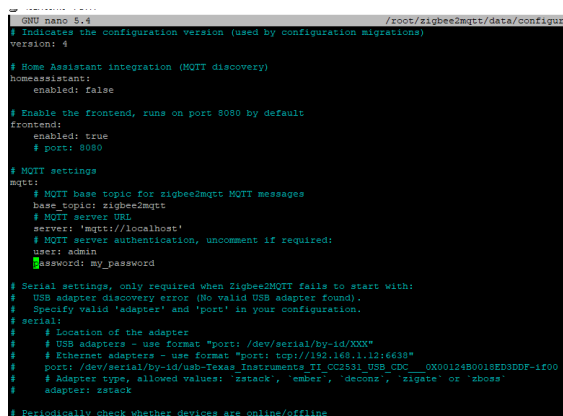
Otwieramy plik do edycji

```
nano /ampio/rw/zigbee2mqtt/data/configuration.yaml
```

Pole *server* ustawiamy na *mqtt://localhost*.

Pole *port* zgodnie z tym co wyszukane zostało powyżej np. */dev/ttyACM0*.

Pola połączenia MQTT zgodnie z naszymi ustawieniami serwera, *user* to najczęściej *admin* oraz odpowiednie hasło (dane logowania jak do blocków *mqtt* w Node-RED).



```
GNU nano 5.4 /root/zigbee2mqtt/data/configuration.yaml
# Indicates the configuration version (used by configuration migrations)
version: 4

# Home Assistant integration (MQTT discovery)
homeassistant:
  enabled: false

# Enable the frontend, runs on port 8080 by default
frontend:
  enabled: true
  port: 8080

# MQTT settings
mqtt:
  # MQTT base topic for zigbee2mqtt MQTT messages
  base_topic: zigbee2mqtt
  # MQTT server URL
  server: 'mqtt://localhost'
  # MQTT server authentication, uncomment if required:
  user: admin
  password: my_password

# Serial settings, only required when Zigbee2MQTT fails to start with:
# USB adapter discovery error (No valid USB adapter found).
# Specify valid 'adapter' and 'port' in your configuration.
serial:
  # Location of the adapter
  # USB adapters - use format "port: /dev/serial/by-id/XXX"
  # Ethernet adapters - use format "port: tcp://192.168.1.12:4638"
  port: /dev/serial/by-id/usb-Texas_Instruments_TI_CC2531_USB_CDC___0801249018E93D0F-if00
  # Adapter type, allowed values: 'Zstack', 'ember', 'deconz', 'zigate' or 'zboss'
  adapter: zstack

# Periodically check whether devices are online/offline
```

Po zmianie zapisujemy i zamykamy plik konfiguracyjny. W putty robimy to poprzez *Ctrl+x*, następnie *y* i *Enter*.

Pierwsze uruchomienie

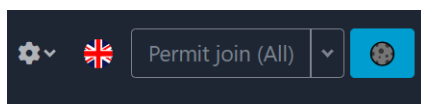
Wpisujemy komendę

```
npm start
```

Dodanie urządzenia podrzędnego

Urządzenie dodawane w poradniku to czujnik temperatury i wilgotności SNZB-02 firmy Sonoff. Instrukcje dla urządzeń dostępne są na stronach odpowiednich producentów. W przypadku opisywanego czujnika, w celu jego dodania należy przytrzymać przycisk na obudowie przez 5 sekund.

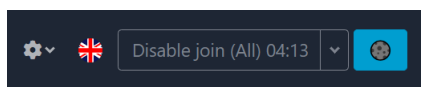
Interfejs Zigbee dostępny jest w przeglądarce pod adresem *IP_SERWERA:8080* (np. *192.168.1.6:8080*). Po wejściu w interfejs możemy dodawać ręcznie kolejne urządzenia poprzez opcję *Permit join*.



Po poprawnym dodaniu urządzenie podrzędne wyświetli się na liście.

#	Pic	Friendly name	IEEE Address	Manufacturer	Model
1		0x00124b00250e039e	0x00124b00250e039e (0xA6B9)	SONOFF	SNZB-02

Na koniec dodawania można jeszcze przedwcześnie zatrzymać dodawanie wszystkiego.



Konfiguracja w Node-RED

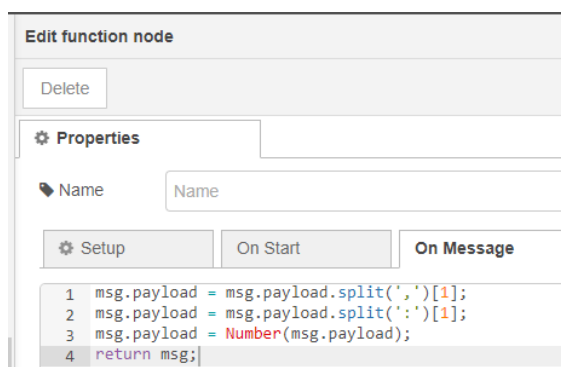
Poradnik opisujący podstawy Node-RED w systemie Ampio dostępny jest pod adresem: [Integracja systemu Ampio z Node-RED](#). Po dodaniu urządzeń podrzędnych można już nasłuchiwać danych z brokera MQTT Ampio. Topic na którym urządzenie nadaje można podejrzeć w terminalu poprzez połączenie SSH. W tym przykładzie jest to topic główny i ID dodanego urządzenia: `zigbee2mqtt/0x00124b00250e039e`.

A screenshot of the 'Edit mqtt in node' configuration window in Node-RED. It has a 'Delete' button, 'Cancel', and 'Done' buttons. The 'Properties' tab is active, showing fields for: Server (localhost:1883), Action (Subscribe to single topic), Topic (zigbee2mqtt/0x00124b00250e039e), QoS (2), Output (auto-detect (string or buffer)), and Name (Name).

Dane można podejrzeć po dodaniu bloczka *debug*.

```
28.02.2023, 09:05:19 node: 3604926b8f956599
zigbee2mqtt/0x00124b00250e039e : msg.payload :
string[85]
"
{"battery":100,"humidity":33.31,"link
quality":220,"temperature":22.97,"vol
tage":3300}"
```

W celu np. odczytania liczbowo wilgotności z tego czujnika przepuszczamy informację przez bloczek *function* z zawartością:



Różne urządzenia końcowe mogą nadawać informację w różnych postaciach dlatego warto podejrzeć dane w oknie *debug* przed napisaniem funkcji przesyłającej informację.

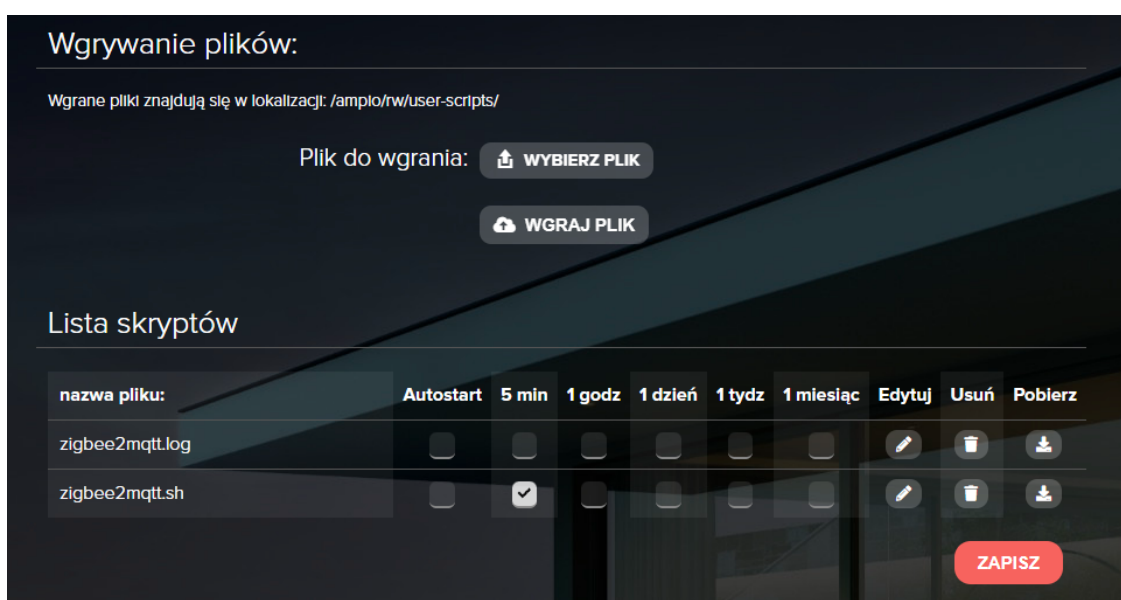
Automatyczne uruchamianie dla obrazów serwera od wersji numer 400

Logujemy się ponownie poprzez SSH, będąc w głównym folderze root pobieramy i uruchamiamy skrypt komendą:

```
curl https://dist.ampio.pl/scripts/zigbee2mqtt400.sh | bash -s
```

dla starszych obrazów

Aby aplikacja została uruchomiona automatycznie po restarcie zasilania, należy dodać odpowiedni skrypt. W interfejsie www modułu M-SERV wchodzimy w zakładkę *SYSTEM* a następnie *SKRYPTY*. Pobieramy załączony plik o nazwie *zigbee2mqtt.sh*. Wgrywamy go poprzez *WYBIERZ PLIK*, następnie *WGRAJ PLIK*. W kolejnym kroku zaznaczamy przy skrypcie pole *5 min* i naciskamy *ZAPISZ*.



Test działania

W celu sprawdzenia poprawności konfiguracji resetujemy zasilanie serwera i po kilku minutach sprawdzamy działanie poprzez Node-RED np. w oknie *debug*.

Jeżeli w trakcie konfiguracji przechodziliśmy w tryb *rw* to na koniec wprowadzamy serwer podczas połączenia SSH ponownie w tryb tylko do odczytu:

```
/opt/ampio/bin/ro
```

Plik do pobrania:

- [zigbee2mqtt.sh](#)