Home Assistant

Home Assistant is een open source thuis automatsering platform

- Installatie
  - ESPHome installeren in Debian 10 (standalone)
  - Home Assistant Core installatie in Debian 10 (Buster) met Python
  - Home Assistant Supervised installatie in Debian 10 (Buster)
- Database
  - SQLite database omzetten naar Postgresql
- Energie
  - Conversie Power Consumption van kW naar W
- ESPHome
- OpenTherm
  - OTGW - OpenTherm Gateway

Installatie

ESPHome installeren in Debian 10 (standalone)

ESPHome is een systeem om een ESP8266/ESP32 microcontroller te configureren met behulp van configuratie bestanden geschreven in YAML. ESPHome maakt zoveel mogelijk gebruik van dezelfde structuur en opbouw van deze YAML bestanden als Home Assistant. ESPHome is daarom ook makkelijk te integreren in Home Assistant.

Je kunt ook gebruik maken van de ESPHome Dashboard, een web interface die het configuratie bestand voor je aanmaakt, controleert op syntax fouten en compileert naar een firmware bestand voor de microcontroler. Dit firmware (.bin) bestand kan je downloaden en daarna in de microcontroller flashen met behulp van je USB naar serieel converter. Eventuele latere wijzigingen van de configuratie kunnen daarna via OTA (Over The Air) weggeschreven worden naar de microcontroller.

Wanneer je Home Assistant Supervised hebt geïnstalleerd, dan kan je op een eenvoudige manier ESPHome installeren en gebruiken vanuit Home Assistant. Heb je Home Assistant Core draaien, dan kan je ESPHome installeren vanaf de commandline.
Dit laatste wordt beschreven in onderstaand artikel.

De installatie instructies lijken in grote lijnen op de **installatie van Home Assistant Core**.
ESPHome is ook geschreven in Python. In dit document gaan we er van uit dat Home Assistant Core reeds is geïnstalleerd.
Voor ESPHome maken we een aparte gebruiker. Ook maken we een virtuele Python omgeving en een systemd opstartscript om de ESPHome Dashboard automatisch te starten wanneer het systeem opstart.

Gebruiker aanmaken voor ESPHome

```
useradd -rm -s /bin/bash esphome -G dialout
```

Virtuele Python omgeving aanmaken

Directory voor de virtuele Python omgeving aanmaken en rechten toekennen:

```
cd /srv
mkdir esphome
chown esphome:esphome esphome
```

Vervolgens gaan we de virtuele Python omgeving maken. Dit doen we onder de gebruiker esphome:

```
sudo -u esphome -H -s
cd /srv/esphome
virtualenv --system-site-packages -p /usr/local/bin/python3.9 /srv/esphome
source bin/activate
```

Nu de virtuele omgeving is aangemaakt kunnen we nog een benodigd Python pakket installeren met:

```
python3 -m pip install wheel
```

En dan installeren we ESPHome:

```
pip3 install esphome
```

Verlaat de virtuele Python omgeving met **exit** of **<CTRL-D>**.

ESPHome Dashboard automatisch starten met systemd

We maken nu een bestand aan zodat ESPHome Dashboard automatisch start bij het opstarten van het systeem.
Open de editor:

```
nano /etc/systemd/system/esphome-dashboard.service
```

Hier plaatsen we het volgende in:

```
[Unit]
Description=ESPHome Dashboard
After=network.target
[Service]
WorkingDirectory=/home/esphome
User=esphome
ExecStart=/srv/esphome/bin/esphome dashboard config
```

```
Restart=on-failure
RestartSec=5s
Environment="PATH=/srv/esphome/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games"
[Install]
WantedBy=multi-user.target
```

In bovenstaand opstartscript zien we dat de **WorkingDirectory** is ingesteld op /home/esphome. Dit betekent dat de configuratie bestanden van ESPHome in deze directory weggeschreven worden.
Deze bestanden staan weer in de directory **config** (dus /home/esphome/config).

De regel met **ExecStart** start de ESPHome Dashboard op.
Je kunt de web interface eventueel ook nog beveiligen met een eigen inlogscherm door onderstaande parameters toe te voegen:

```
ExecStart=/srv/esphome/bin/esphome dashboard config --username user --password password --open-ui
```

De regel die begint met **Environment** zorgt er voor dat de omgevingsvariabelen goed ingesteld worden.
In dit geval is dat het pad naar de juiste directories. Wanneer dit niet goed staat ingesteld, werkt het compileren niet in ESPHome.

Vervolgens voeren we het volgende uit:

```
systemctl --system daemon-reload
systemctl enable esphome-dashboard.service
systemctl start esphome-dashboard.service
```

Controleer of de service nu actief is met:

```
systemctl status esphome-dashboard.service
```

Je kunt nu naar de ESPHome Dashboard via http://<IP-ADRES-MACHINE>:6052

ESPHome updaten

Wanneer er een nieuwe versie uit is van ESPHome kan men deze via de commandline updaten:

```
sudo -u esphome -H -s
cd /srv/esphome
source bin/activate
pip3 install --upgrade esphome
```

Als dit succesvol is verlopen kan je de virtuele omgeving verlaten met <CTRL-D> of exit. Hierna dient men ESPHome te herstarten met:

```
systemctl restart esphome-dashboard.service
```

Bronnen

ESPHome homepage

How to create a systemd service for python script with virtualenv

Home Assistant Core installatie in Debian 10 (Buster) met Python

In dit document wordt beschreven hoe je Home Assistent Core kan installeren door middel van Python in Debian 10 (Buster).
Home Assistant Core bevat dus niet de Supervisor en de add-ons.

Python installeren

Op het moment van schrijven is voor Home Assistant Python 3.9 nodig. In Debian 10 hebben we standaard alleen de beschikking over Python 2.7 en Python 3.7. Je kan met onderstaand commando controleren welke versies van Python geïnstalleerd zijn in Debian:

```
ls /usr/bin/python*
```

De actieve Python versie kan je controleren met:

```
python -V
```

Hoe je de nieuwste Python 3.9 versie kan installeren vanuit source staat beschreven in het artikel Python installeren vanuit source in Debian 10.

Hoe je de standaard Python versie kan instellen en kan wisselen van Python versie staat beschreven in het artikel Default Python versie wijzigen in Debian 10.

Overige benodigde pakketten installeren

Onderstaande uitvoeren om de rest van de benodigde pakketten te installeren:

```
apt-get install libffi-dev libssl-dev libjpeg-dev zlib1g-dev autoconf build-essential
libopenjp2-7 libtiff5 sudo
```

Gebruiker aanmaken voor Home Assistant

We gaan nu een gebruiker aanmaken speciaal voor Home Assistant:

```
useradd -rm -s /bin/bash homeassistant -G dialout
```

Bovenstaand commando maakt de gebruiker homeassistant aan, maakt de home directory /home/homeassistant aan en voegt de gebruiker toe aan de groep dialout.

Virtuele Python omgeving aanmaken

Je kunt in Python een virtuele omgeving maken die onafhankelijk draait van de rest. Het voordeel is dat je eventueel verschillende Python versies naast elkaar kan draaien. Voor Home Assistant maken we een virtuele omgeving aan in de directory /srv/homeassistant.
De configuratie bestanden van Home Assistant komen te staan in /home/homeassistant.

We gaan eerst het Python pakket pip updaten (pip is de package installer voor Python) en vervolgens het pakket virtualenv voor het maken van de virtuele Python omgeving. **We gaan in onderstaande handelingen uit van een geïnstalleerde Python 3.9 ve**rsie.

```
python3.9 -m pip install --upgrade pip
pip3 install virtualenv
```

We voeren het volgende uit om de directory aan te maken en de gebruiker homeassistant eigenaar te maken van deze directory:

```
cd /srv
mkdir homeassistant
chown homeassistant:homeassistant homeassistant
```

Vervolgens gaan we de virtuele Python omgeving maken. Dit doen we onder de gebruiker homeassistant.
Let op dat je de juiste python versie kiest, in ons geval python3.9.
Je kunt het juiste pad controleren op de commandline met: which python3.9

```
sudo -u homeassistant -H -s
cd /srv/homeassistant
virtualenv --system-site-packages -p /usr/local/bin/python3.9 /srv/homeassistant
source bin/activate
```

Merk op dat we in de virtuele omgeving zitten door de prompt die je nu ziet.
In mijn geval is dat **(homeassistant) homeassistant@hass:/srv/homeassistant$**
Nu de virtuele omgeving is aangemaakt kunnen we nog een benodigd Python pakket installeren met:

```
python3 -m pip install wheel
```

En uiteindelijk installeren we Home Assistant Core met:

```
pip3 install homeassistant
```

Start Home Assistant Core nu eenmalig door:

```
hass
```

De installatie wordt nu afgemaakt en de configuratie directory /home/homeassistant/.homeassistant wordt ook aangemaakt.
Je kunt nu met je webbrowser naar http://<IP-ADRES-MACHINE>:8123 om de eerste gebruiker aan te maken voor Home Assistant.
Let op dat het even kan duren voordat je dit scherm krijgt. Home Assistant is dan nog niet klaar met de installatie, dus heb even geduld.

Wanneer je kan inloggen en Home Assistant werkt, dan kan je het nu afbreken met **Control-C**.
Verlaat de virtuele Python omgeving nu met **exit** of **<CTRL-D>**.

Home Assistant Core automatisch starten met systemd

We maken nu een bestand aan zodat Home Assistant automatisch start bij het opstarten van de PC.
Open de editor:

```
nano /etc/systemd/system/home-assistant@homeassistant.service
```

Hier plaatsen we het volgende in:

```
[Unit]
Description=Home Assistant
After=network-online.target

[Service]
Type=simple
User=%i
WorkingDirectory=/home/%i/.homeassistant
ExecStart=/srv/homeassistant/bin/hass -c "/home/%i/.homeassistant"

[Install]
```

```
WantedBy=multi-user.target
```

Vervolgens voeren we het volgende uit:

```
systemctl --system daemon-reload
systemctl enable home-assistant@homeassistant.service
systemctl start home-assistant@homeassistant.service
```

Controleer of de service nu actief is met:

```
systemctl status home-assistant@homeassistant.service
```

Home Assistant updaten

Er komen regelmatig nieuwe updates uit van Home Assistant. De update kan vanaf de commandline uitgevoerd worden binnen de virtuele Python omgeving. We doen dit onder de gebruiker homeassistant en gaan daarna naar de virtuele omgeving:

```
sudo -u homeassistant -H -s
cd /srv/homeassistant
source bin/activate
```

Controleer of je ook daadwerkelijk in de virtuele omgeving zit door de juiste prompt die je krijgt:
(homeassistant) homeassistant@hass:/srv/homeassistant$
Daarna kunnen we Home Assistant updaten met het commando:

```
pip3 install --upgrade homeassistant
```

Als dit succesvol is verlopen kan je de virtuele omgeving verlaten met <CTRL-D> of exit.
Hierna dient men Home Assistant te herstarten met:

```
systemctl restart home-assistant@homeassistant.service
```

Automatisch naar virtuele omgeving gaan voor homeassistant

Om niet iedere keer bovenstaande handelingen uit te hoeven voeren als je Home Assistant wilt updaten kunnen we het bestand ~/.bashrc gebruiken in de home directory van homeassistant. Hiervoor gaan we eerst naar de home directory toe:

```
cd /home/homeassistant
```

Onderstaande commando’s zorgen ervoor dat er twee regels toegevoegd worden in het bestand .bashrc die er voor zorgen dat we direct in de virtuele omgeving zitten zodra we inloggen met de gebruiker homeassistant:

```
sh -c 'echo "cd /srv/homeassistant" >> ~/.bashrc'
sh -c 'echo "source /srv/homeassistant/bin/activate" >> ~/.bashrc'
```

Log nu in als gebruiker homeassistant met:

```
sudo -u homeassistant -H -s
```

Je zou nu in de virtuele omgeving moeten zitten, herkenbaar aan de prompt.

Bronnen

Home Assistant – Installation

Autostart using systemd

Updating Home Assistant

Home Assistant Supervised installatie in Debian 10 (Buster)

In deze instructies wordt beschreven hoe men Home Assistant Supervised kan installeren in Debian 10 (Buster).
De installatie zal in grote lijnen ook uitgevoerd kunnen worden in andere Linux distributies.

Home Assistant Supervised is een complete installatie van alle modules (Home Assistant Core, Home Assistant Supervisor en add-ons).
Alle modules draaien in Docker. De installatie wordt uitgevoerd door middel van een script.

Debian updaten en benodigde pakketten installeren

Zorg er eerst voor dat Debian up-to-date is:

```
apt-get update
apt-get upgrade
```

Installeer vervolgens de volgende benodigde pakketten:

```
apt-get install software-properties-common apparmor-utils apt-transport-https ca-certificates
curl dbus jq network-manager
```

ModemManager uitschakelen

ModemManager hebben we niet nodig en kan conflicten geven met eventuele andere devices, dus gaan we deze uitschakelen:

```
systemctl disable ModemManager
systemctl stop ModemManager
```

Docker installeren

```
curl -fsSL get.docker.com | sh
```

Home Assistant Supervised installeren

```
curl -sL "https://raw.githubusercontent.com/Kanga-Who/home-assistant/master/supervised-installer.sh" | bash -s
```

De installatie heeft enige tijd nodig.
Je kunt nu met je webbrowser gaan naar http://<IP-ADRES-MACHINE>:8123
Wanneer de installatie klaar is krijg je het inlogscherm te zien. Je kan dan een eerste inlog account aanmaken.
Hierna kan je verder gaan met de configuratie van Home Assistant.

Bronnen

https://community.home-assistant.io/t/installing-home-assistant-supervised-on-debian-10/200253

Database

Database

SQLite database omzetten naar Postgresql

Standaard maakt Home Assistant gebruik van een SQLite database. Wanneer er veel informatie wordt weggeschreven is het verstandig om te kiezen voor een wat robuustere oplossing als database. In dit artikel wordt beschreven hoe je de bestaande SQLite database kan omzetten naar een Postgresql database en deze actief maakt in Home Assistant.

Benodigde pakketten installeren

Eerst installeren we de benodigde Debian pakketten:

```
apt-get install postgresql postgresql-server-dev-13 pgloader
```

Postgresql gebruiker maken voor homeassistant

Nu gaan we eerst een gebruiker maken binnen Postgresql speciaal voor de database die we gaan gebruiken voor Home Assistant:

```
sudo -u postgres createuser homeassistant
```

Negeer de melding: “could not change directory to “/root”: Permission denied”.

Vervolgens kennen we een wachtwoord toe voor de gebruiker (role) homeassistant:

```
su - postgres
# psql -c "ALTER USER homeassistant WITH PASSWORD 'securepass_here';"
```

Postgres database maken voor homeassistant

Nu maken we de database aan voor Home Assistant.

```
sudo -u postgres createdb -O homeassistant homeassistant
```

Hier wordt de database met de naam homeassistant gemaakt en de eigenaar is de gebruiker homeassistant.

Negeer wederom de melding: “could not change directory to “/root”: Permission denied”.

Python pakket psycopg2 installeren in de virtuele omgeving

Om Python te laten communiceren met Postgresql moeten we een pakketje installeren genaamd psycopg2.

Dit doen we in ons geval binnen de virtuele Python omgeving waar Home Assistant draait.
Eerst zorgen we dat we binnen de virtuele omgeving werken:

```
sudo -u homeassistant -H -s
cd /srv/homeassistant/
source bin/activate
```

Je moet nu in de virtuele omgeving zitten. Dit zie je aan de prompt:

```
(homeassistant) homeassistant@hass:/srv/homeassistant$
```

Vervolgens gaan we psycopg2 installeren binnen de virtuele omgeving:

```
pip3 install psycopg2
```

Wanneer dit succesvol is verlopen kan je de virtuele omgeving verlaten met <CTRL-D> of exit. Je komt dan weer in de normale omgeving terecht.

Home Assistant configuratie aanpassen

Nu gaan we de Home Assistant configuratie aanpassen zodat de Postgresql database gebruikt wordt in plaats van de SQLite database.
Open hiervoor het bestand configuration.yaml, in de editor. In ons geval is dat:

```
nano /home/homeassistant/.homeassistant/configuration.yaml
```

Plaats het volgende in dit configuratie bestand:

```
recorder:
  db_url: postgresql://@/homeassistant
```

De recorder integratie binnen Home Assistant is verantwoordelijk voor de opslag van gegevens in de database.

Herstart vervolgens Home Assistant om de nieuwe configuratie actief te maken:

```
systemctl restart home-assistant@homeassistant.service
```

Controleer de status met:

```
systemctl status home-assistant@homeassistant.service
```

Als alles goed is gegaan mogen er geen fouten optreden. Mocht er iets niet werken, bekijk dan de logfiles van Home Assistant en/of Postgresql. Home Assistant werkt nu met een “lege” Postgresql database. In onderstaande stappen gaan we de bestaande SQLite data overzetten naar de Postgresql database.

SQLite database omzetten naar Postgresql

Het overzetten van de SQLite database naar Postgresql doen we met de tool genaamd pgloader. We maken eerst een commando bestand aan die we straks aanroepen met pgloader. In dit bestand staan o.a. de regels vanuit welke database de gegevens gehaald moeten worden en in welke Postgresql database ze weggeschreven moeten worden. Het bestand slaan we even op onder de naam pglcommand. Open de editor:

```
nano pglcommand
```

Hier plaatsen we de volgende regels in:

```
load database
    from '/home/homeassistant/.homeassistant/home-assistant_v2.db'
    into postgresql://homeassistant:securepass_here@localhost/homeassistant

with data only, drop indexes, reset sequences, truncate

set work_mem to '16MB', maintenance_work_mem to '512 MB';
```

Pas bovenstaand bestand aan voor jouw situatie. Dus het juiste pad naar de Home Assistant database (home-assistant_v2.db) en de juiste Postgresql gebruiker met bijbehorend wachtwoord (homeassistant:securepass_here). In bovenstaande situatie gaan we er van uit dat de Postgresql machine lokaal draait en de database heet homeassistant (localhost/homeassistant).

Wanneer dit bestand is opgeslagen gaan we Home Assistant stoppen met:

```
systemctl stop home-assistant@homeassistant.service
```

Vervolgens gaan we de gegevens overzetten naar de Postgresql database met:

```
pgloader pglcommand
```

Pas bovenstaand commando eventueel aan met het juiste pad naar pglcommand.

Je zou nu een overzicht moeten krijgen met de tabellen die aangemaakt zijn en het aantal items die overgezet zijn. Zie je geen tabellen en staan de items op 0, controleer dan het pglcommand bestand.

Start vervolgens Home Assistant weer op met :

```
systemctl start home-assistant@homeassistant.service
```

Foutmeldingen na omzetten: “duplicate key value violates unique constraint”

Na het omzetten van de database constateerde ik veel foutmeldingen in de logfile van Postgresql (/var/log/postgresql/postgresql-13-main.log) en de logfile van Home Assistant. Deze foutmeldingen hadden allemaal dezelfde melding: **duplicate key value violates unique constraint**. Het betrof de onderstaande tabellen in Postgresql:

- statistics_short_term
- statistics_runs
- statistics

Dit bleek te maken te hebben met de waarden in de “id” kolom van de desbetreffende tabellen. De waarde die in deze kolom weggeschreven moet worden blijkt dan al te bestaan. Het lijkt er dus op dat deze “id” kolom niet leeg wordt gemaakt met het omzetten van de SQLite naar de Postgresql database. In mijn ogen lijkt dit toch wel te moeten gebeuren met het uitvoeren van pgloader. Hier staat immers de optie “reset sequences” in het commando bestand. Om het probleem op te lossen voeren we onderstaande SQL commando’s uit in de psql omgeving. Log dus eerst in als postgres gebruiker:

```
su - postgres
```

En ga vervolgens naar de psql omgeving:

```
psql
```

Je zit nu in de juiste omgeving. Vervolgens moeten we opgeven met welke database we willen gaan werken. In ons geval is dat de database “homeassistant”. Dit doen we met het volgende commando:

```
\c homeassistant
```

Je krijgt dan de melding “You are now connected to database “homeassistant” as user “postgres”. Hierna kunnen onderstaande SQL commando’s één voor één uitgevoerd worden. Dus na iedere

regel een <ENTER> om het commando uit te voeren:

```
SELECT setval('statistics_short_term_id_seq', (SELECT MAX(id) FROM statistics_short_term)+1);
SELECT setval('statistics_runs_run_id_seq', (SELECT MAX(run_id) FROM statistics_runs)+1);
SELECT setval('statistics_id_seq', (SELECT MAX(id) FROM statistics)+1);
```

Deze commando’s zorgen er voor dat de hoogste waarde in de kolom “id” of “run_id” wordt opgezocht van de desbetreffende tabellen. Vervolgens wordt deze waarde met 1 opgehoogd en weggeschreven in de bijbehorende sequence tabel. Je zou hierna geen foutmeldingen meer mogen krijgen.

Om alle tabellen te bekijken in de homeassistant database kan je het volgende commando uitvoeren:

```
\d
```

Je krijgt dan onderstaand overzicht:

```
                        List of relations
 Schema |             Name              |   Type   |     Owner
--------+-------------------------------+----------+---------------
 public | events                        | table    | homeassistant
 public | events_event_id_seq           | sequence | homeassistant
 public | recorder_runs                 | table    | homeassistant
 public | recorder_runs_run_id_seq      | sequence | homeassistant
 public | schema_changes                | table    | homeassistant
 public | schema_changes_change_id_seq  | sequence | homeassistant
 public | states                        | table    | homeassistant
 public | states_state_id_seq           | sequence | homeassistant
 public | statistics                    | table    | homeassistant
 public | statistics_id_seq             | sequence | homeassistant
 public | statistics_meta               | table    | homeassistant
 public | statistics_meta_id_seq        | sequence | homeassistant
 public | statistics_runs               | table    | homeassistant
 public | statistics_runs_run_id_seq    | sequence | homeassistant
 public | statistics_short_term         | table    | homeassistant
 public | statistics_short_term_id_seq  | sequence | homeassistant
(16 rows)
```

Wil je zien welke kolommen er in een tabel zijn, dan kan je onderstaand commando uitvoeren:

```
\d <tabelnaam>
```

Energie

Energie

Conversie Power Consumption van kW naar W

In Home Assistant maak ik gebruik van de integratie "DSMR Slimme Meter" om mijn energie en gas te registreren.
De entiteit "Power Consumption" laat standaard zijn waarden zien in kilo Watt (kW). Deze wil ik graag naar Watt (W) hebben.

Om dit voor elkaar te krijgen maken we gebruik van een Home Assistant template. Hiervoor moeten we een stukje aanpassen in het configuratie bestand van Home Assistant:

configuration.yaml.

Open het configuratie bestand in de editor en voeg het volgende toe:

```
template:
  - sensor:
      - name: "Huidig Vermogenverbruik"
        unit_of_measurement: "W"
        state: >
          {{(states('sensor.power_consumption') | float * 1000) | round(0)}}
```

Sla dit op en herstart Home Assistant:

```
systemctl restart home-assistant@homeassistant.service
```

Je hebt hierna een entiteit er bij gekregen met de naam "Huidig Vermogenverbruik". Deze kan je weer toevoegen in je dashboard.
De gemeten waarden worden nu weergegeven in Watt (W).

ESPHome

OpenTherm

OTGW - OpenTherm Gateway

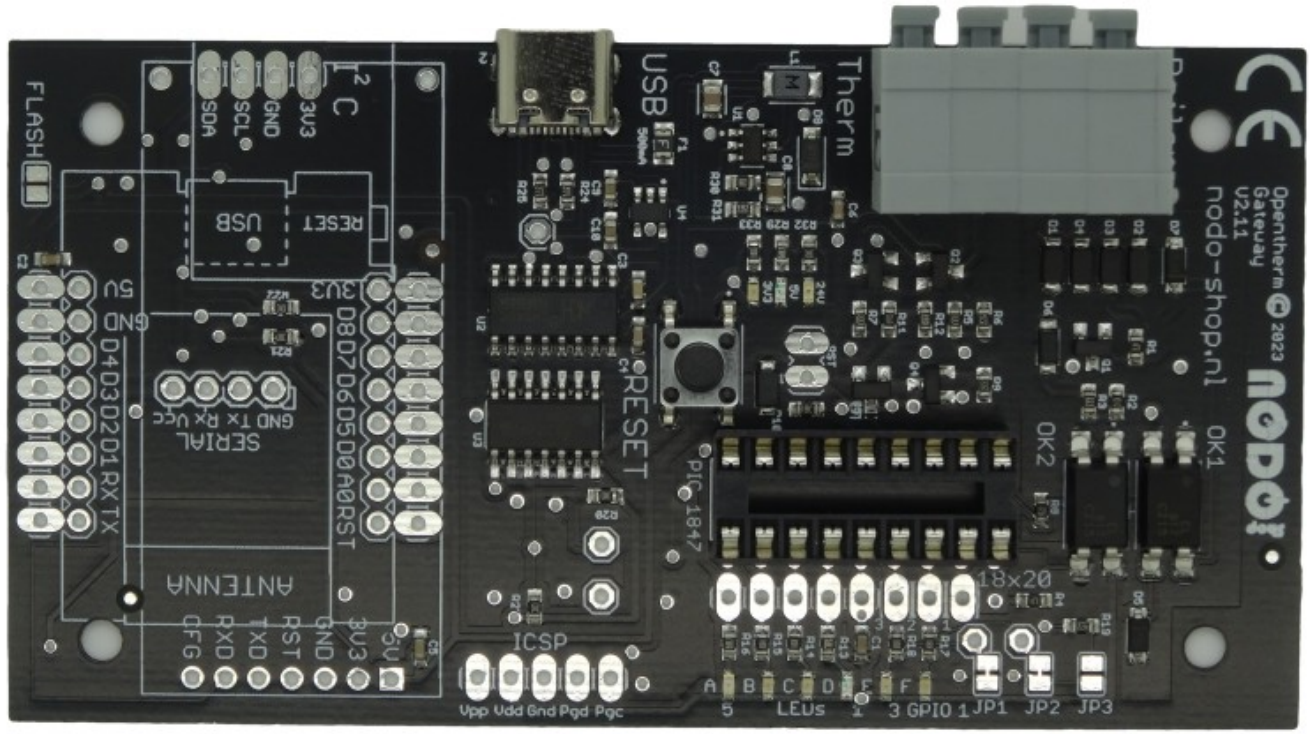
OpenTherm is het protocol wat veel CV-ketels gebruiken om met de thermostaat te communiceren. Een OpenTherm gateway zit tussen de CV-ketel en de thermostaat en kan o.a. het OpenTherm protocol monitoren.
Ook kan je je temperatuur instellen met behulp van deze gateway.

Er zijn diverse elektronica projecten te vinden op het internet hoe zo'n OpenTherm Gateway te bouwen.
Ook kan je een complete gateway aanschaffen die je deels nog zelf moet solderen.

Deze is bijvoorbeeld verkrijgbaar bij de Nodo Shop en is gebaseerd op een opensource project gemaakt door Schelte Bron.

Op de site van Schelte Bron vindt u alle informatie over deze gateway http://otgw.tclcode.com ,

daarnaast is er ook veel informatie te vinden op: Domotica OTGW forum.



Opentherm Monitor

Op de website van Schelte Bron vind je ook een stukje software die kan communiceren met de gateway, genaamd

de Opentherm Monitor (otmonitor). Deze tool kan je ook zonder GUI (headless) laten draaien in Linux.
Hiervoor kan je een configuratiebestand maken, bijvoorbeeld otmonitor.conf, die je dan als parameter meegeeft
op de commandline. Hier onder volgt een voorbeeld van het configuratiebestand:

```
web {
  enable true
  port 8080
  nopass true
}
connection {
  enable true
  type tcp
  host <IP-adres OpenTherm Gateway>
  port <poortnummer>
}
mqtt {
  enable true
  broker <IP-adres mqtt broker>
  port <poort mqtt>
  username <mqtt username>
  password <mqtt password>
  devicetype central_heating
  deviceid otmonitor
  format unformatted
}
server {
  enable true
  port 7686
  relay true
}
clock {
  year true
  date true
  auto true
}
```

In bovenstaand voorbeeld wordt er gebruik gemaakt van een TCP verbinding via W-Fi (Wemos D1 mini) of de vaste LAN aansluiting (USR-TCP232-T2 module). De connection optie "host" is dan het IP-adres van de OpenTherm Gateway met de bijbehorende poort (optie "port").

Maak je gebruik van de seriële aansluiting, dan dien je de "connection" optie als volgt in te stellen:

```
connection {
  device /dev/ttyUSB0
  type serial
  enable true
}
```

Er van uitgaande dat hier dus een USB naar serieel adapter gebruikt wordt (/dev/ttyUSB0).

Maak je geen gebruikt van MQTT, dan kan de sectie "mqtt" weggelaten worden.
Hetzelfde geldt voor de sectie "server". Dit heb je alleen nodig wannneer je otmonitor als relay server gebruikt.

Vervolgens kan je otmonitor als volgt starten op de commandline:

```
./otmonitor-x64 --daemon -f /opt/otmonitor/otmonitor.conf
```

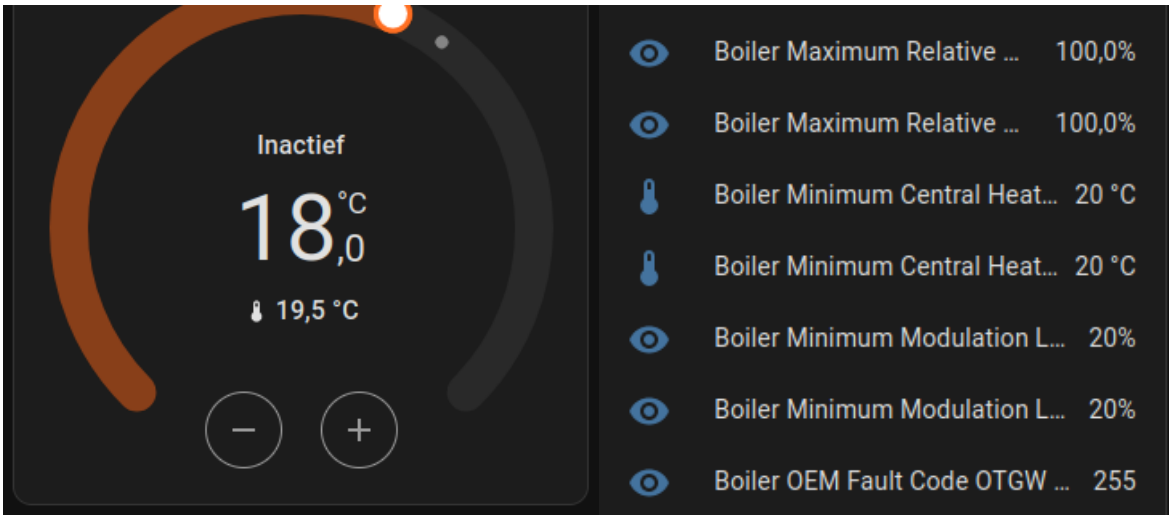
In bovenstaand voorbeeld staat het configuratiebestand dus in de directory /opt/otmonitor.
De optie --daemon geeft aan dat otmonitor zonder gui moet starten op de achtergrond.
Meer commandline opties vind je met:

```
./otmonitor-x64 --help
```

Je kunt nu met je webbrowser gaan naar http://<IP-adres>:8080 om de web GUI te benaderen van otmonitor.
Je hebt hier dezelfde mogelijkheden als via de normale GUI.

Home Assistant - OpenTherm Gateway integratie

In Home Assistant bestaat er een integratie OpenTherm Gateway die je kunt gebruiken in combinatie met bovenstaande hardware van de Nodo Shop. Je hebt dan de beschikking over een een Climate Control waar je de temperatuur mee in kan stellen.



Verder heb je heel veel entiteiten beschikbaar met de diverse waarden en instellingen van de CV-ketel.
Met behulp van automatiseringen kan je bijvoorbeeld ervoor zorgen dat de temperatuur lager gezet wordt wanneer er niemand thuis is. Wanneer er dan iemand thuis komt mag de thermostaat het weer overnemen en verder gaan met het klokprogramma.
Hier onder een voorbeeld van de automatiseringen die ik zelf gebruik:

CV naar 15 graden wanneer niemand thuis is:

```
alias: CV naar 15 graden wanneer niemand thuis is
description: ""
trigger:
  - platform: state
    entity_id:
      - device_tracker.samsungalex
    to: not_home
  - platform: state
    entity_id:
      - device_tracker.samsungali
    to: not_home
condition:
  - condition: and
    conditions:
      - condition: state
        entity_id: device_tracker.samsungalex
        state: not_home
      - condition: state
```

```
        entity_id: device_tracker.samsungali
        state: not_home
action:
  - service: climate.set_temperature
    data:
      temperature: 15
    target:
      device_id: 40a23f7ed1c456c6dbcba8ea8037b074
mode: parallel
max: 10
```

CV verder met klok programma thermostaat:

```
alias: CV verder met klok programma thermostaat
description: ""
trigger:
  - platform: state
    entity_id:
      - device_tracker.samsungalex
    from: not_home
    to: home
  - platform: state
    entity_id:
      - device_tracker.samsungali
    from: not_home
    to: home
condition:
  - condition: or
    conditions:
      - condition: state
        entity_id: device_tracker.samsungalex
        state: not_home
      - condition: state
        entity_id: device_tracker.samsungali
        state: not_home
action:
  - service: climate.set_temperature
    target:
      device_id: 40a23f7ed1c456c6dbcba8ea8037b074
    data:
```

```
      temperature: 0
mode: parallel
max: 10
```

In bovenstaande automatisering wordt de temperatuur op "0" gezet. Hiermee wordt de waarde "Remote override room setpoint" van 15 graden weer ongedaan gemaakt en neemt de thermostaat het weer over met het ingestelde klokprogramma.