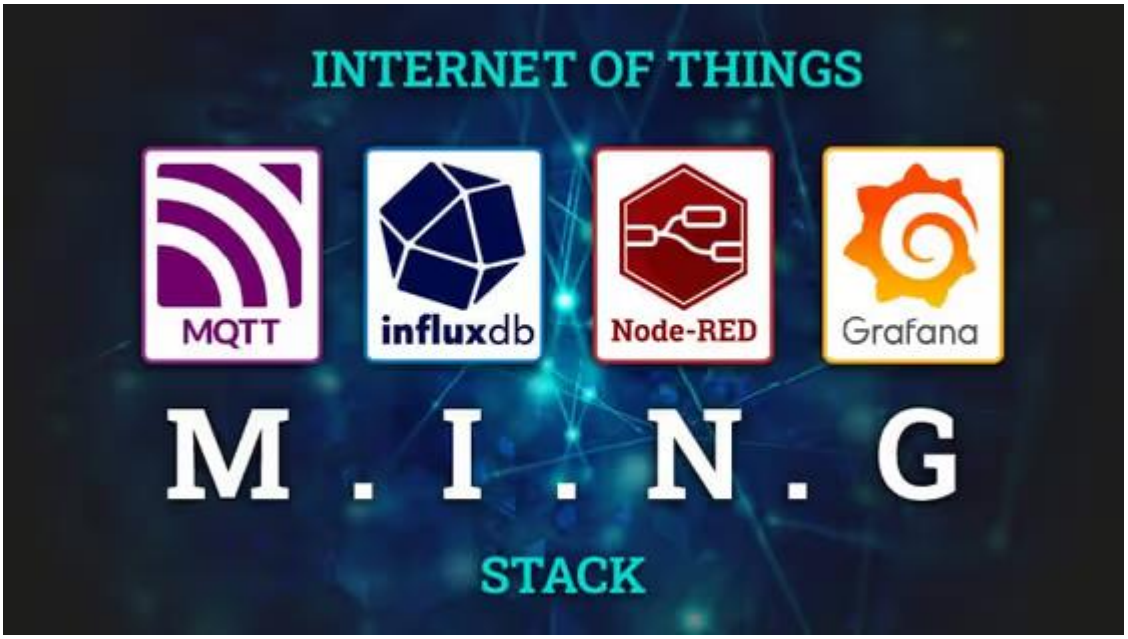


IoT Prosjekt

Server oppsett

Innhold

1 Hva er MING – stack

Her har vi brukt chat-gpt til å oversette og tilpasse en artikkel fra FlowFuse [1]

MING-stakken, som er forkortelse for Mosquitto/MQTT, InfluxDB, Node-RED og Grafana, representerer et sett med teknologier som er optimalisert for utvikling av IoT-applikasjoner.

1. **Mosquitto (MQTT-broker)**: Dette er en populær og åpen kildekode-MQTT-broker. MQTT (Message Queuing Telemetry Transport) er et lettvekts, pub/sub-basert protokoll som er spesielt egnet for IoT-applikasjoner. Mosquitto muliggjør pålitelig kommunikasjon mellom IoT-enheter ved å håndtere meldingsformidling.
2. **InfluxDB (tidsseriedatabase)**: InfluxDB er en kraftig, åpen kildekode-database optimalisert for å håndtere tidsseriedata. Dette er spesielt viktig for IoT-scenarioer der man trenger å lagre og analysere data som endres over tid, for eksempel sensoravlesninger eller strømforbruk. InfluxDB muliggjør effektiv lagring og henting av slik data.
3. **Node-RED (lavkodeutviklingsmiljø)**: Node-RED er et visuelt programmeringsverktøy som forenkler utviklingen av IoT-applikasjoner ved å tillate enkel tilkobling og konfigurasjon av forskjellige enheter og tjenester. Det gir et grafisk grensesnitt for å opprette flyter av data mellom forskjellige enheter og skytjenester, noe som er spesielt nyttig i komplekse IoT-økosystemer.
4. **Grafana (visualiseringsplattform)**: Grafana er et kraftig verktøy for dataanalyse og visualisering. Det gir mulighet for å opprette detaljerte og interaktive dashbord som kan vise sanntidsdata fra IoT-enheter på en forståelig måte. Dette gjør det enkelt for utviklere og sluttbrukere å overvåke ytelse, identifisere trender og ta informerte beslutninger.

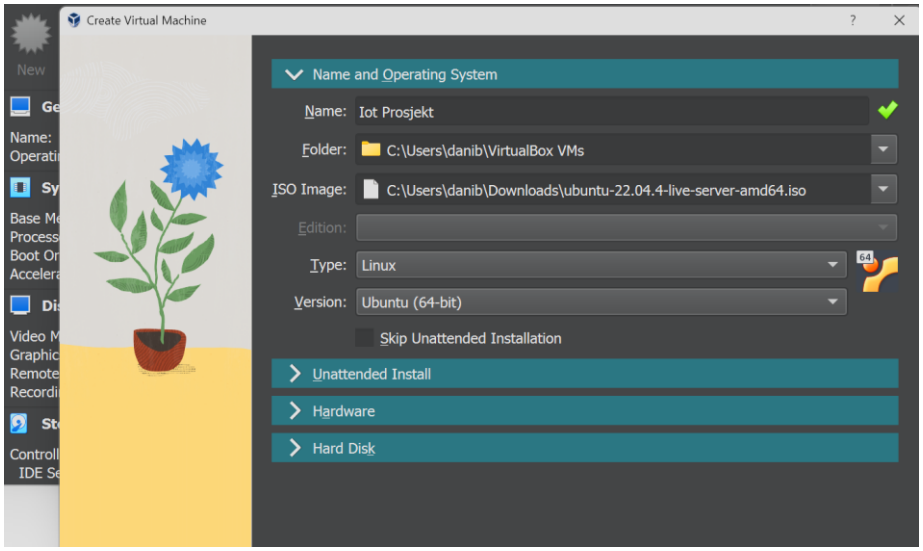
MING-stakken tilbyr en komplett løsning for utvikling av IoT-applikasjoner ved å kombinere kommunikasjon, databehandling og visualisering på en integrert måte. Ved å bruke disse teknologiene sammen, kan utviklere raskt prototypere, implementere og skalere komplekse IoT-systemer, enten de er skybaserte eller distribuert på kanten av nettverket.

2 Sett opp server

Før vi starter op med MING-stack må vi ha en Linuxserver programmene skal kjøre på. Dette kan gjøres enten virtuelt eller på egen server.

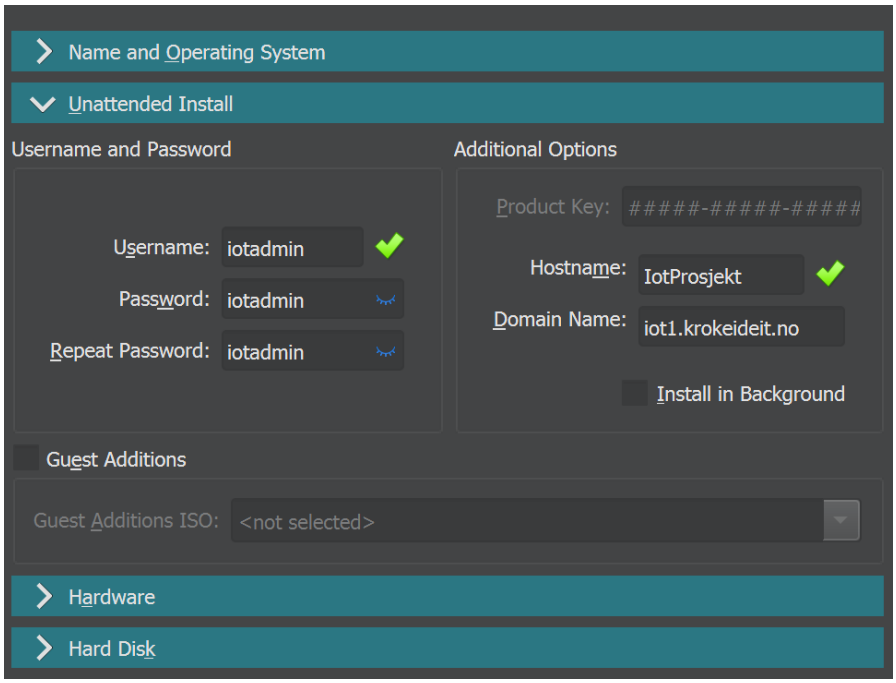
2.1 Sett opp VM i Virtual Box:

1) Klikk “New”
2) Velg navn på den virtuelle maskinen, hvor den skal lagres, hvilken bildefil og type operativsystem.

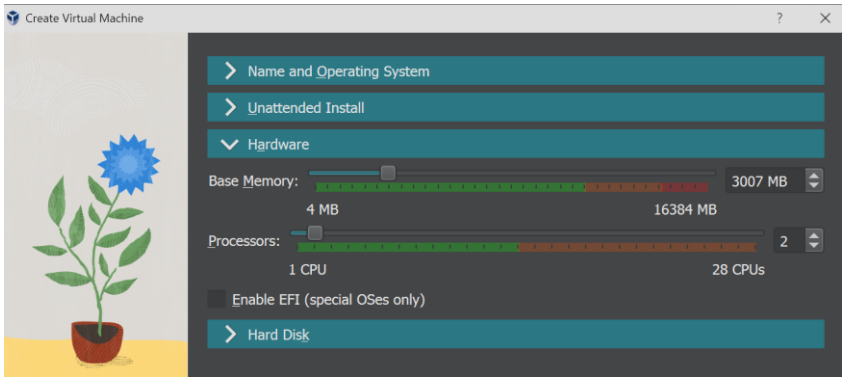


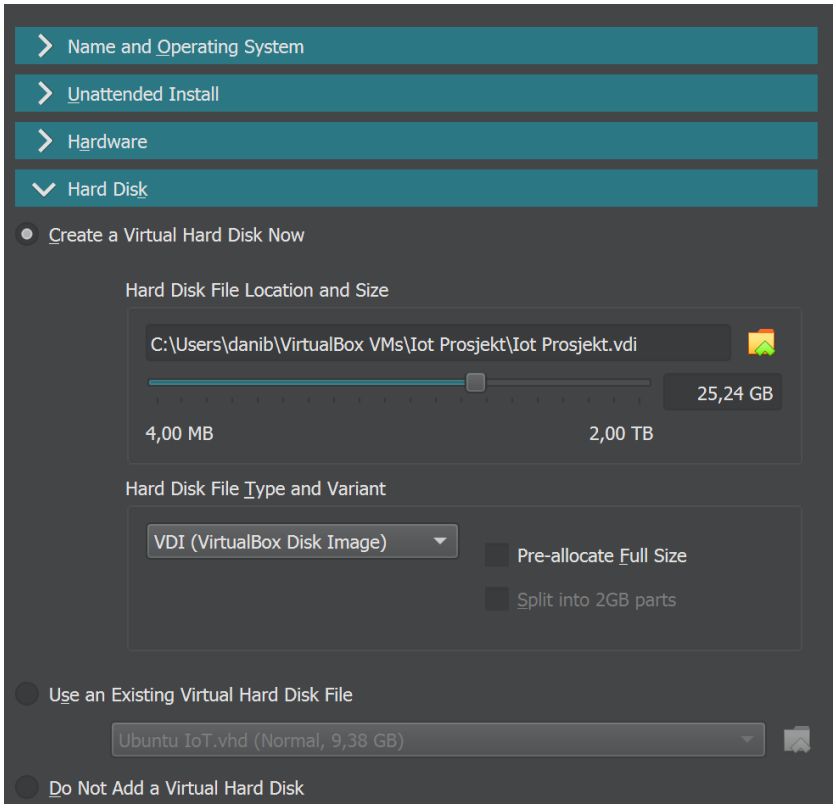
3) Unattended Install konfigurerer litt av installasjonen for deg

Her velger vi brukernavn, passord, og “hostname” (navnet på maskinen som vi kan bruke i stedet for IP-adresse). Domenet er ikke så viktig men skolen har noen domener som kan benyttes.



4) Gi ressurser (CPU, RAM, HDD) til den virtuelle maskinen og trykk Finish:

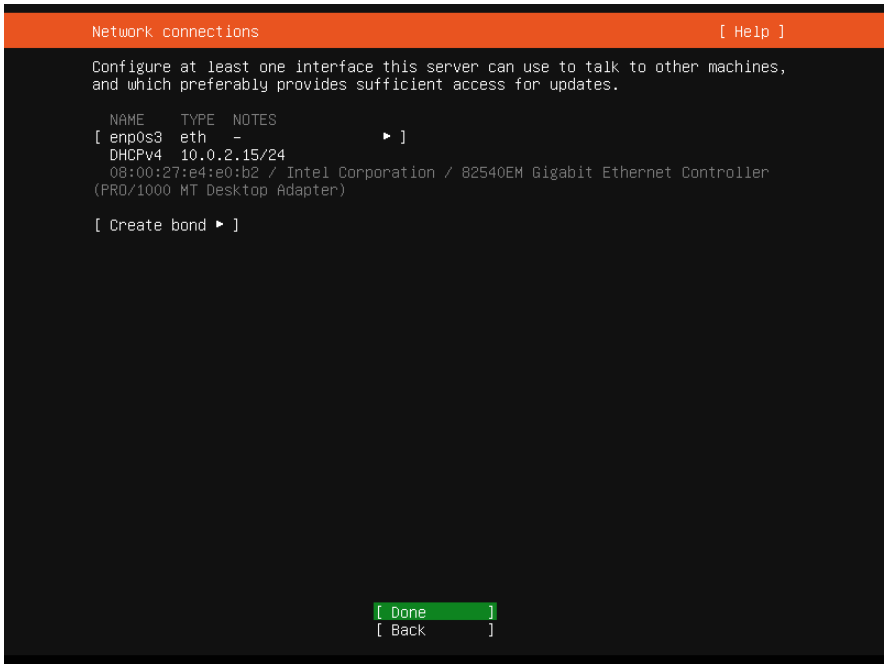




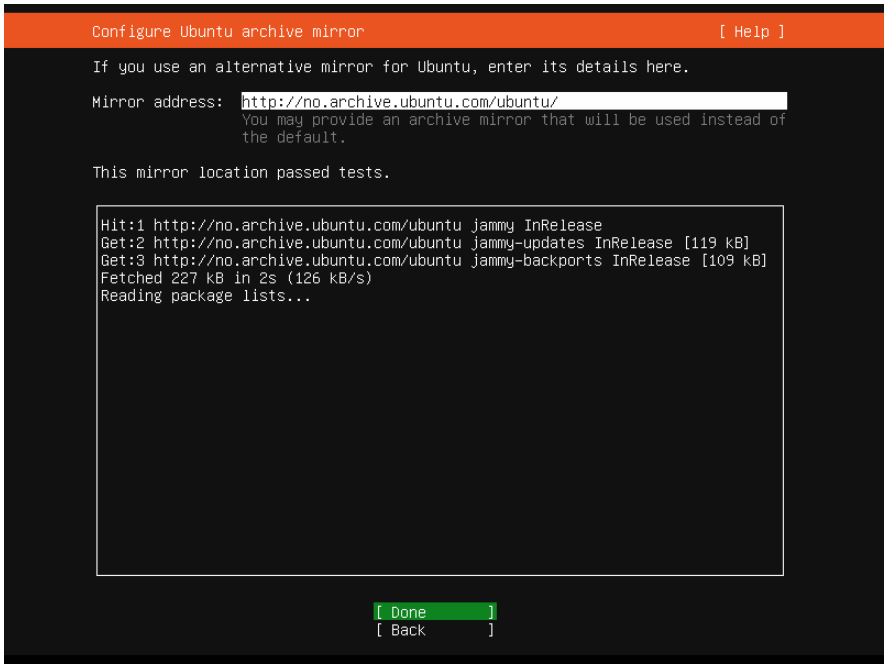
.

2.2 Installer Ubuntu:

1) Velg “English” språk -> som utviklere kan norsk lage krøll.
2) Velg “Norwegian” tastatur og layout
3) Velg vanlig “Ubuntu Server”, har du Nvidia skjermkort velg også “third-party drivers”
4) Trykk done når installasjonen har funnet nettverkskortet ditt:

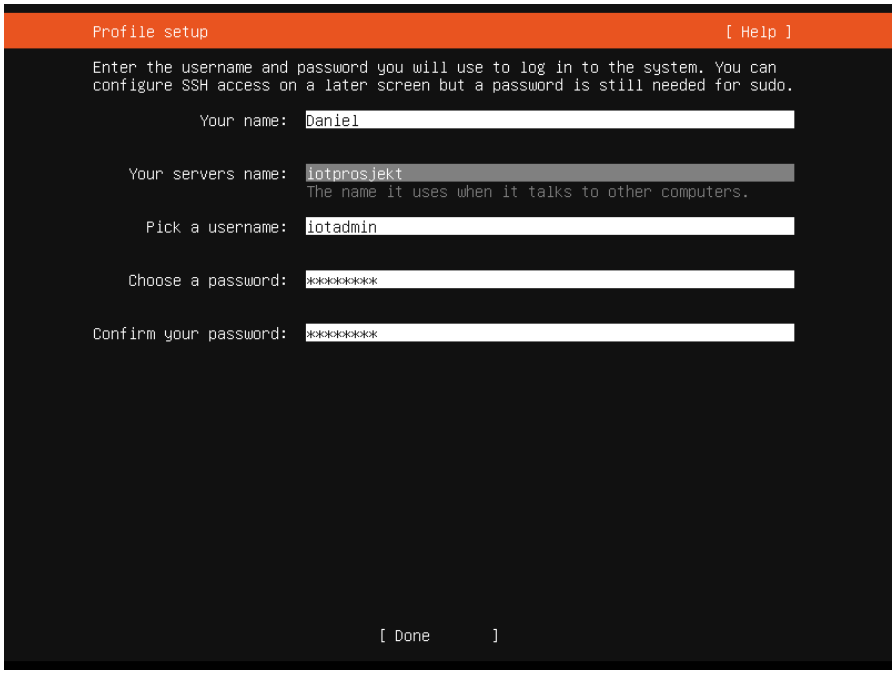


5) Vi benytter ikke proxy
6) Vent til “mirror location” har “passed test”. Dette er serveren filene skal hentes fra.



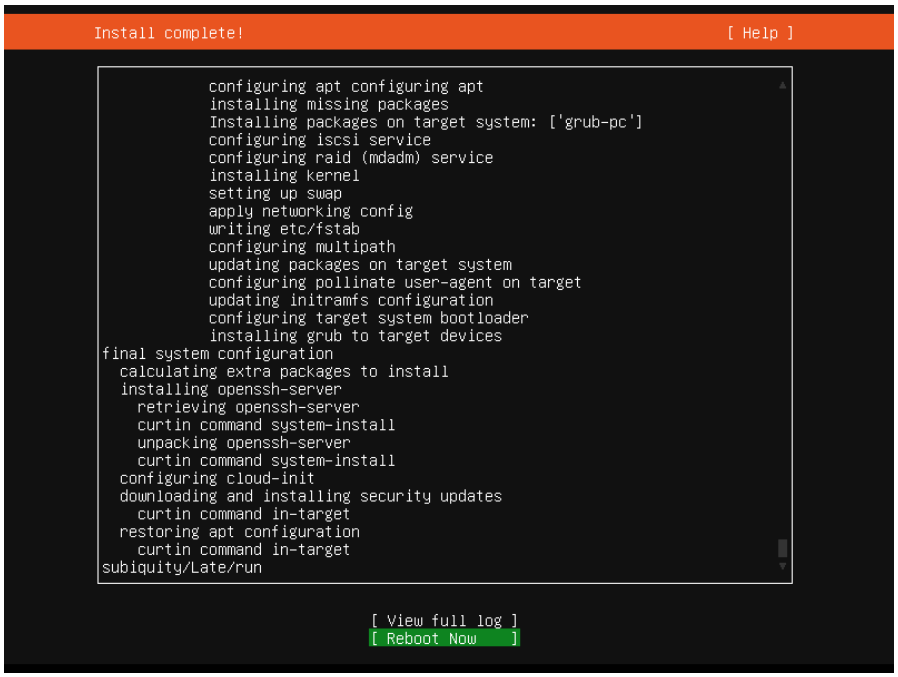
7) Trykk Done gjennom disk oppsettet. Altså bruk hele disken som LVM uten kryptering.

8) Sett opp brukeren og servernavnet:

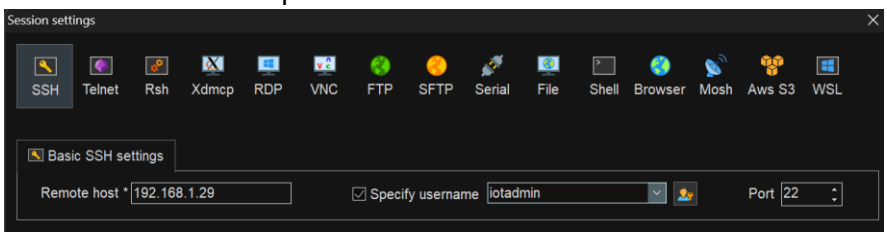


9) Installer OpenSSH, men ingenting annet.

10) Trykk reboot når du har valget:



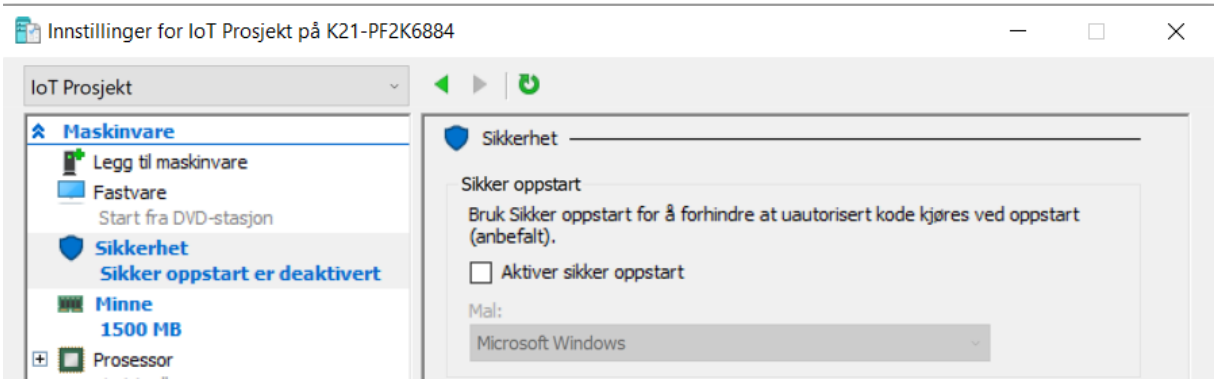
11) Nå kan du velge om du vil sette opp resten lokalt, eller logge deg på med SSH og konfigurere fra en annen PC på det samme nettverket. MobaXterm er anbefalt for ssh. Finn IP-en din med: ‘ip addr show’:



2.3 Sett opp server med Hyper-V

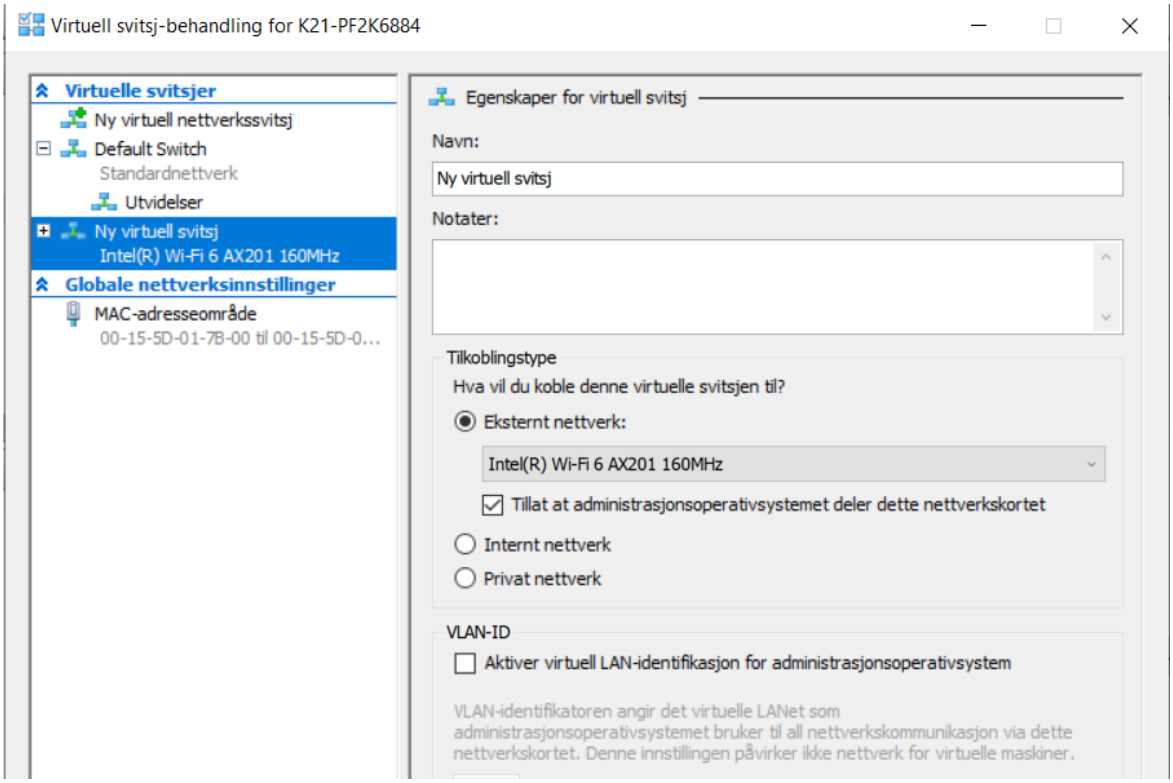
Følg oppsett i Linux prosjektet i IT-kompendium

Skru av Windows sikker oppstart

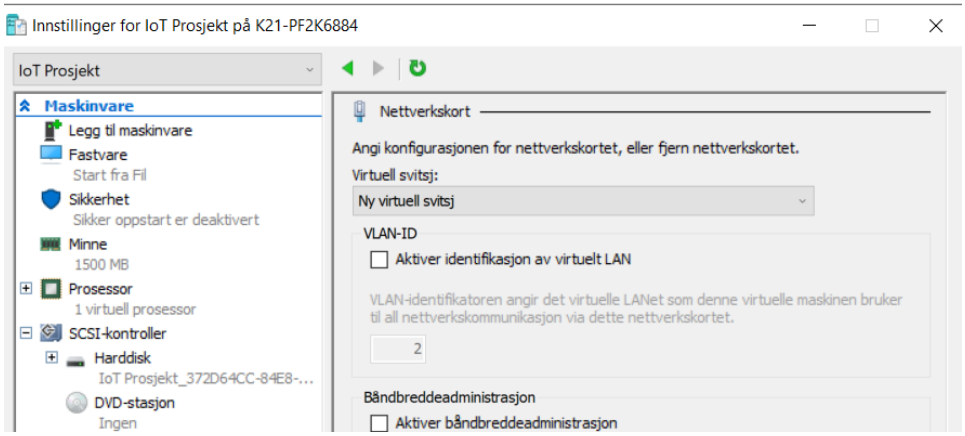


Lag en default switch som bruker samme nettverkskort som pc som «bridger» VM til samme kort som pc.

Default oppsett er ved nat tilkobling mellom vm og pc.



Gå på VM og velg “ny virtuell svitsj”

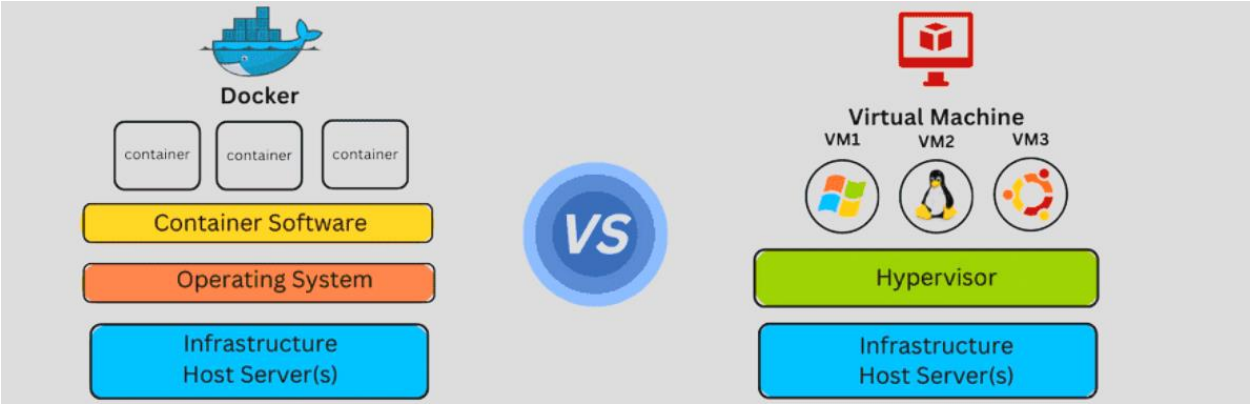

Innstillinger for IoT Prosjekt på K21-PF2K6884
IoT Prosjekt
Maskinvare
Legg til maskinvare
Fastvare
Start fra Fil
Sikkerhet
Sikker oppstart er deaktivert
Minne
1500 MB
Prosessor
1 virtuell prosessor
SCSI-kontroller
Harddisk
IoT Prosjekt_372D64CC-84E8-...
DVD-stasjon
Ingen
Nettverkskort
Angi konfigurasjonen for nettverkskortet, eller fjern nettverkskortet.
Virtuell svitsj:
Ny virtuell svitsj
VLAN-ID
Aktiver identifikasjon av virtuelt LAN
VLAN-identifikatoren angir det virtuelle LANet som denne virtuelle maskinen bruker til all nettverkskommunikasjon via dette nettverkskortet.
2
Båndbreddeadministrasjon
Aktiver båndbreddeadministrasjon

3 MING-konteinere i Docker på virtuell Ubuntu server

Docker	-	Virtualiseringsprogram som setter opp “konteinere” som inneholder virtuelle maskiner. Hver konteiner kjører ett program, dermed blir det ikke noe krøll med annen installert software.
Docker Compose	-	Docker Compose lar oss kjøre skript som automatisk setter opp konteinere. Den leser en .yml fil og setter opp konteinerne vi ønsker.
Portainer	-	Portainer lar oss administrere aller docker konteinerne vi har med et grafisk grensesnitt.

Til høyre på bildet ser vi VirtualBox serveren vår: PC-en er host/infrastruktur, så ligger hypervisoren vår VirtualBox som lager VM med ubuntu. VirtualBox er type 2 hypervisor så den kjører inne i windows (ikke vist på tegningen)

Til venstre ser vi Docker. Docker lager “konteinere” som er virtuelle maskiner for hver tjeneste. Hver konteiner inneholder nødvendige biblioteksfiler for å kjøre tjenesten.



Konteinerne vi lar Docker sette opp er en MING-stack. Det gir oss alt vi trenger for en komplett IoT server:

M	-	MQTT	-	Protokoll for å overføre sensor data
I	-	InFluxdb	-	Database eller “excel-ark” for å lagre data fra sensor
N	-	Node-Red	-	Visuelt programmeringsspråk som knytter alt sammen
G	-	Grafana	-	Tar data fra databasen og presenterer de i grafer

3.1 Sett opp Docker på din server[2]:

Kjør kommandoer direkte i terminal eller gjennom SSH med f.eks MobaXterm.

1) Vi oppdaterer alltid listen over software i apt "butikken" og oppdaterer systemet med:

```
sudo apt update && sudo apt upgrade -y
```

2) Oppdater apt-get "butikken" og Legg til nøkler slik at "butikken" kan hente fra Docker:

```
sudo apt-get update

sudo apt-get install ca-certificates curl
```

3) For å laste ned Docker må vi legge Docker sin repository til i apt «butikken» vår. Da trenger vi nøkler som lagres i keyrings. Lag en mappe for keyrings og last ned nødvendige nøkler.

Kjør denne koden **linje for linje fra** https://docs.docker.com/engine/install/ubuntu/

```
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/d
sudo chmod a+r /etc/apt/keyrings/docker.asc
```

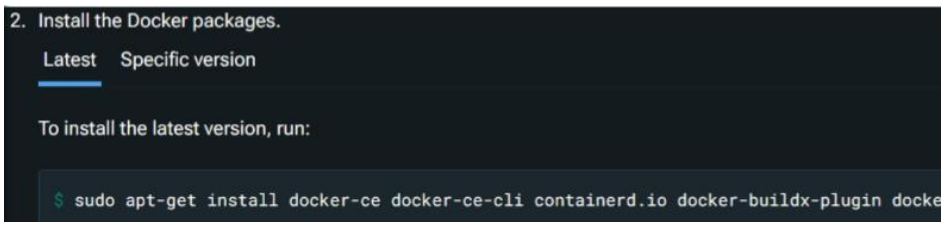
4) Legg Docker repository til i apt ved å kjøre koden Kjør denne koden **fra** https://docs.docker.com/engine/install/ubuntu/

```
# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] h
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

5) Oppdater apt listen slik at Docker repositorien blir inkludert

```
sudo apt update
```

6) Legg til plugins (som docker-compose) fra: https://docs.docker.com/engine/install/ubuntu/



7) Test Docker installasjonen med:

```
sudo docker run hello-world
```

3.2 Sette opp MING-konteinere med Docker Compose

Nå skal vi installere programmene i Ming-stack

1) Lag en mappe for å ha docker-compose.yml og mosquitto.config filene i

```
sudo mkdir iot
```

2) gå inn i mappen og last ned en ferdig .yml fil for å sette opp dokker, og konfig til MQTT.

```
cd iot

wget techne.guru/docker-compose.yml
```

3) inspiser yml filen med nano. Ctrl+X tar deg ut av programmet. Sudo lar deg redigere.

```
nano docker-compose.yml
```

4) Kjør docker-compose.yml filen. Nå vil Docker opprette ferdige konteinere med program:

```
sudo docker compose up -d
```

5) Konfigurer mosquitto ved å laste ned konfigurasjonsfilen og brukerdata til rett sted:

```
cd ~/iot/volumes/mosquitto/config/

sudo wget techne.guru/mosquitto.conf && sudo wget techne.guru/passwords.txt
```

6) Passwords.txt filen er generert med kommandoen under. Her er brukernavn iotadmin. For å lage din egen password fil med et annet brukernavn, kjør kommandoen og bytt iotadmin.

```
Sudo docker run -it --rm -v
/iot/volumes/mosquitto/config:/mosquitto/config eclipse-mosquitto
mosquitto_passwd -c /mosquitto/config/passwords.txt iotadmin
```

7) Nå er alt satt opp. Restart serveren med

```
sudo shutdown now -r
```

<u>Serveren kjører nå MING programmene i hver sine konteinere:</u>

Konteinerne kan administreres fra en hvilken som helst nettleser med Portainer.

1) Du har en **M**QTT broker som tar imot sensordata fra ESP32-S3.
2) Disse lagres i en database som heter sensor_data på **I**nfluxDB
3) **N**ode Red er klart for å programmere dataflyten
4) **G**rafana er klart for å lage dashbord med grafer for dataene fra sensoren.

3.3 MING-Konteiner liste

Nå er alle programmene installert. Videre bruker vi en nettleser for å konfigurere oppsettet.

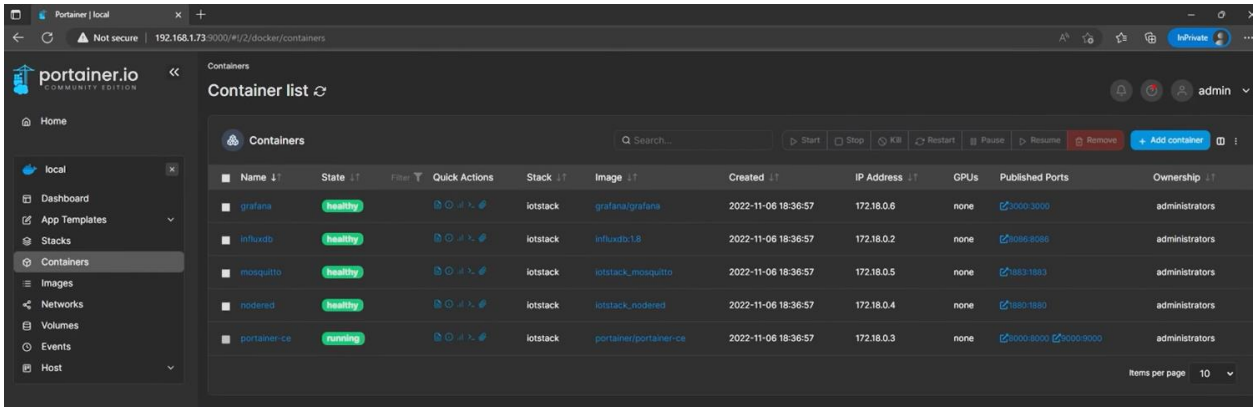
Tabellen nedenfor viser hvilke porter de ulike programmene bruker.

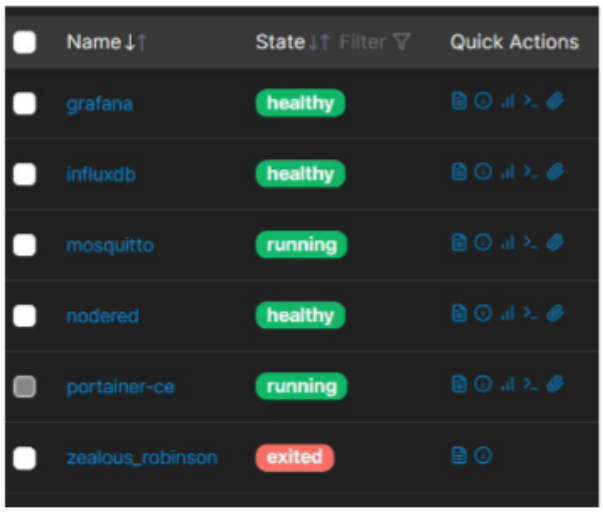
Tjeneste	Port	Brukernavn	Passord	Beskrivelse
InfluxDB	ingen	`admin`	`iotadmin1234`	Database
Grafana	`3000`	`admin`	`iotadmin1234`	Visualiseringsverktøy `http://<din-ip>:3000`.
Mosquitto	`1883` (MQTT), `9001` (Websocket)	iotadmin	iotadmin	MQTT broker for å sende og motta data til ESP32-S3
Node-RED	`1880`	Ikke satt opp	Ikke satt opp	Programmerer dataflyten `http://<din-ip>:1880`.
Portainer	`9000`	`admin`	Settes opp ved login	Docker management verktøy. `http://<din-ip>:9000`

4 Sjekk konteinere med Portainer

1) Du kan kontrollere tilstanden på konteinerne ved å gå inn på Portainer GUI gjennom en hvilken som helst nettleser på det samme nettverket.

   Adressen er: http://<din-IP>:9000





Navnene her kan du bruke i stedet for IP-addresser. Dersom serveren får ny IP-addresse slipper du å bytte IP på alle konteinerne!

Quick Actions har nyttige knapper. "logs" gir deg live logger på det som skjer i konteineren.

5 Node-RED:

Vi skal nå sette opp Node-RED for å håndtere data som kommer inn til serveren.

Se denne videoen som forklarer oppsettet[3]: https://youtu.be/ffg3_1AgtyA?t=779

Første delen av videoen viser oppsett av server på en Raspberry PI, men det har vi allerede gjort på vår egen server.

Han går videre gjennom oppsett av Node-RED, Influx og Grafana.

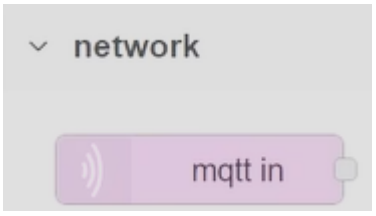
Eventuelt kan du følge oppskriften nedenfor

1) Gå inn på Node-RED GUI gjennom en nettleser på det samme nettverket.

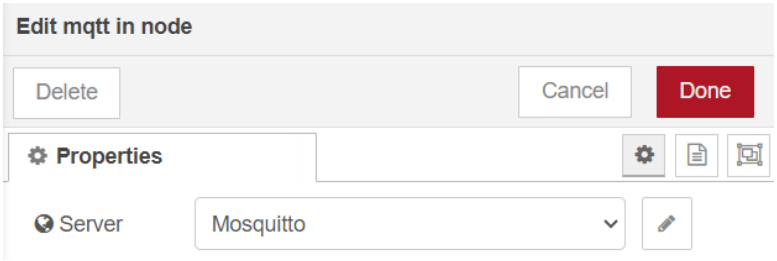
   Adressen er http://<din-IP>:1880.

   Siden vi har virtualisert serveren kan du bruke nettleseren på laptopen din utenfor virtualbox.

2) Trekk en MQTT input til skrivebordet:

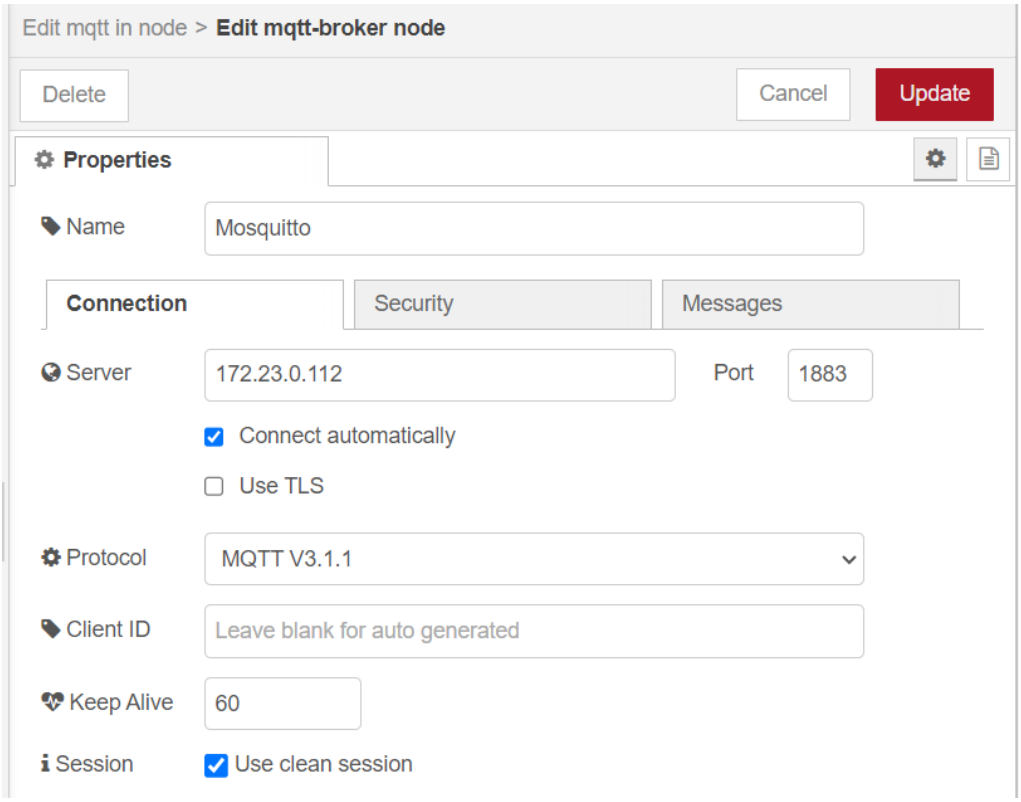


3) Vi må fortelle Node-RED hvor MQTT serveren er, ved å 'add new mqtt-broker'.

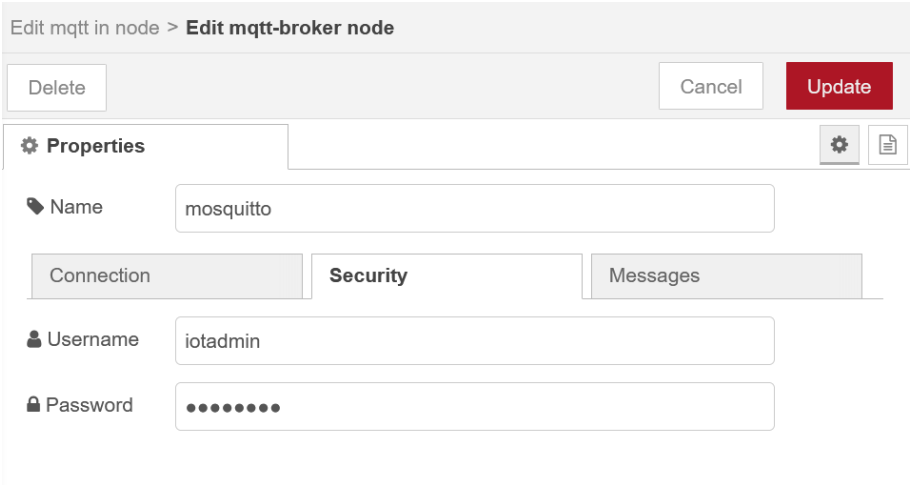


Trykk på blyant til høyre for Server

Der skriver vi inn vår egen IP-adresse



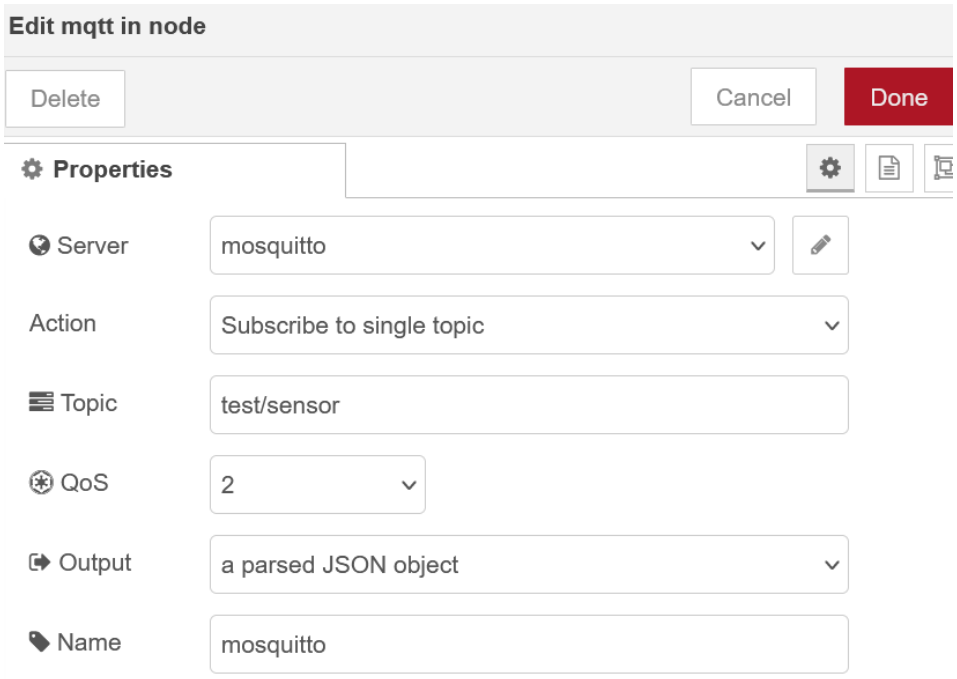
Velg Security for å gi serveren et navn og brukernavn/passord:



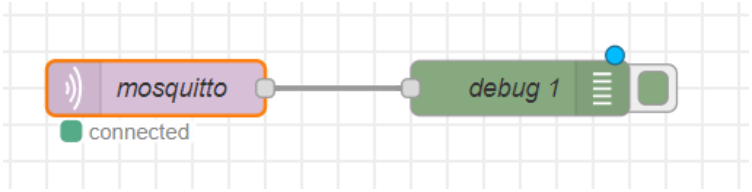
4) Det er viktig å ha rett Topic.

Vi har benyttet én topic for temperatur og fuktighet for koden i ESP32-S3.

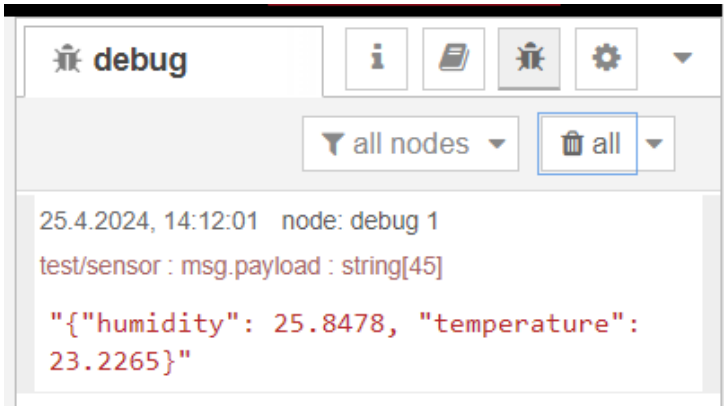
Derfor skriver vi **test/sensor** i topic



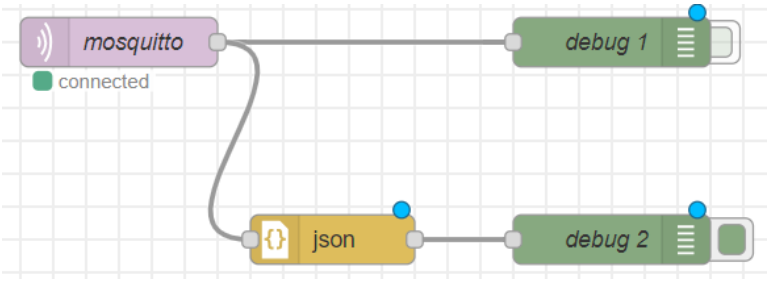
5) Legg inn en «debug» blokk og aktiver denne



6) Velg debug i meny til høyre og du vil se meldinger fra noden din

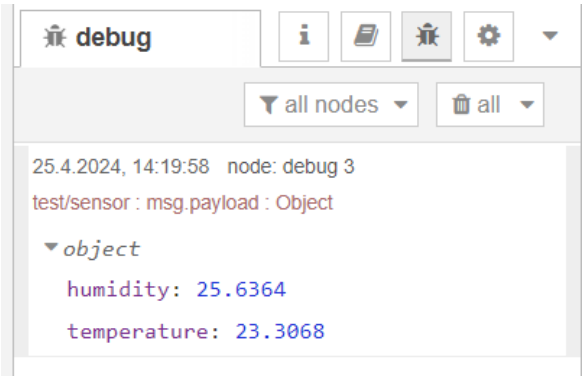


7) Koble en Json blokk til mosquitto. Legg også til en debug blokk.

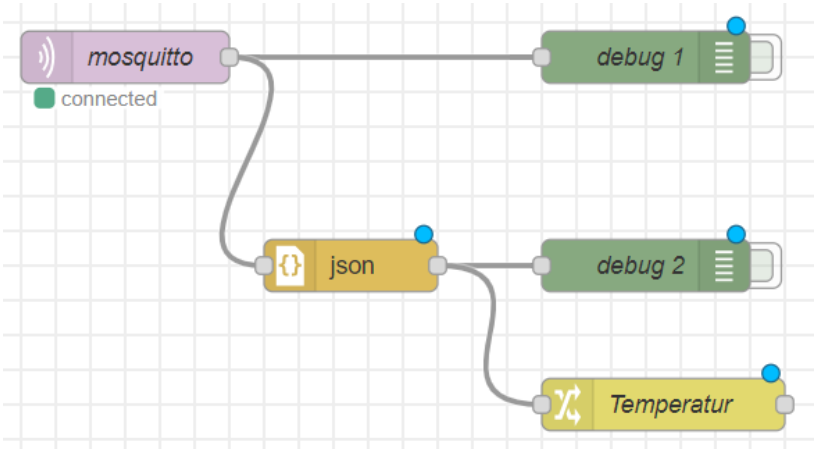


8) Aktiver debug 2 og vent på innkommende melding fra node.

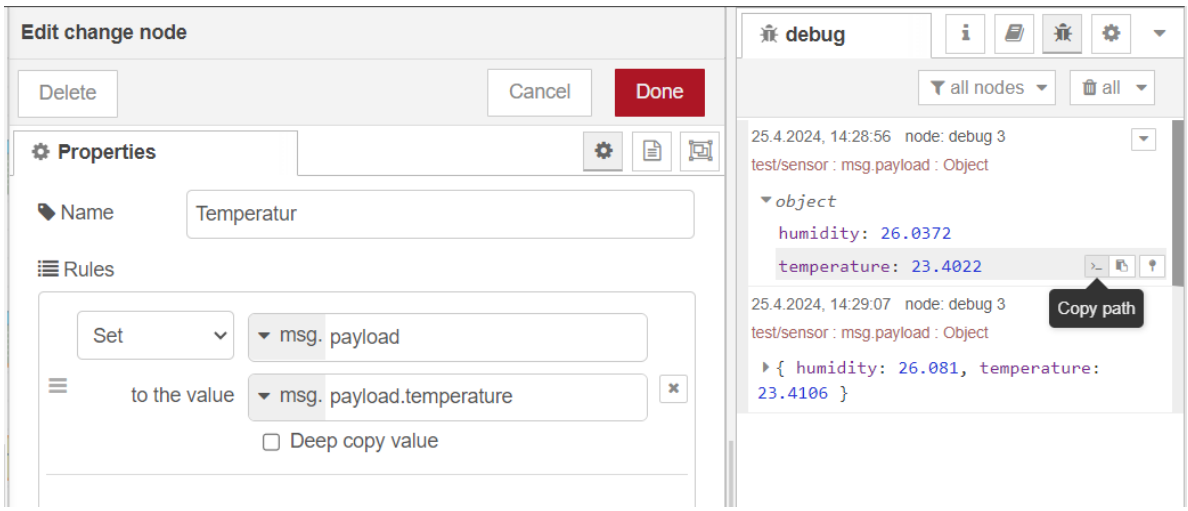
Nå er det blitt laget et Json objekt som har to data; humidity og temperatur



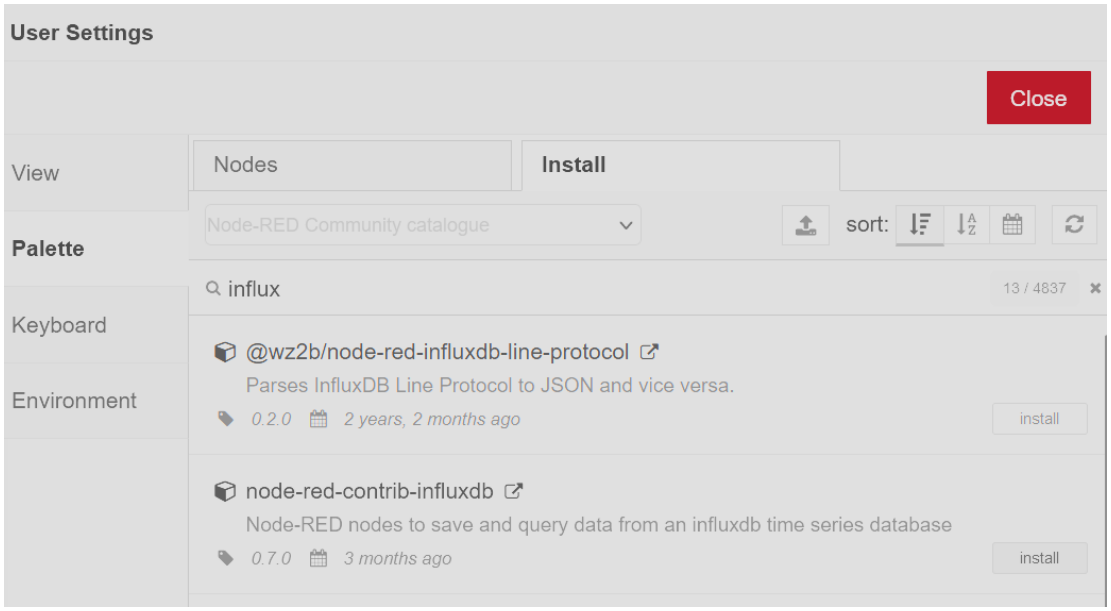
9) For å trekke ut enkeltdata bruker vi en «change» blokk



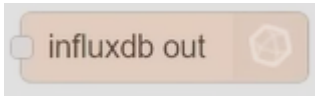
10) Set utgangsverditype til msg og bruk path fra debugvindu til riktig data



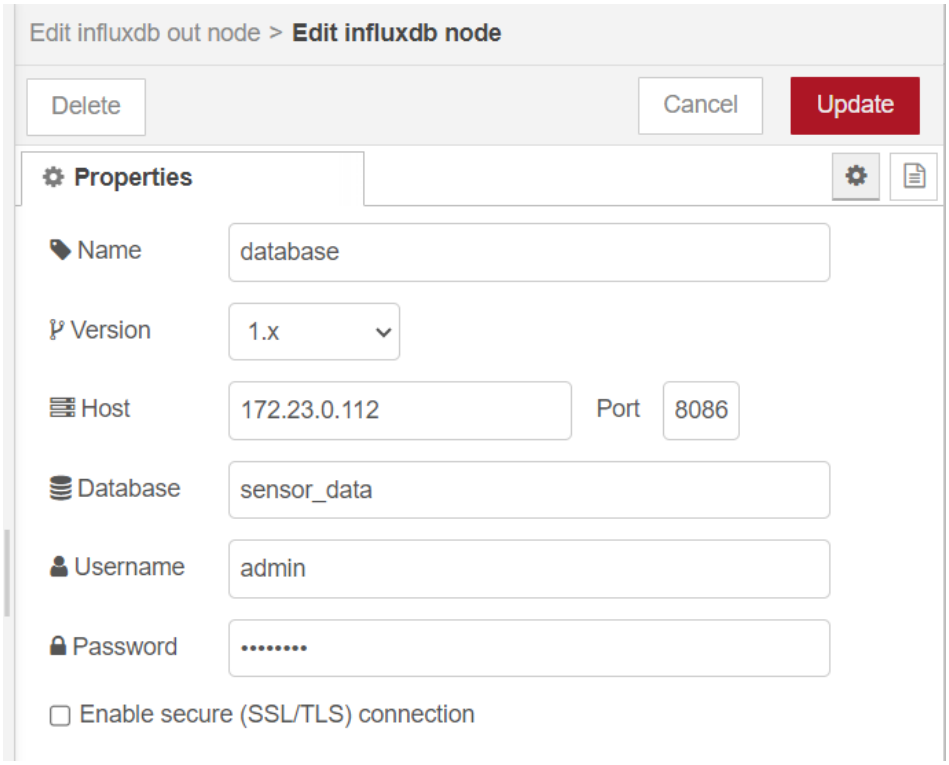
11) Vi må installere en blokk for influxdb. Velg ManagePalette i meny til høyre eller trykk: shift+alt+p og Installer node-red-contrib-influxdb plugin for influxdb funksjoner



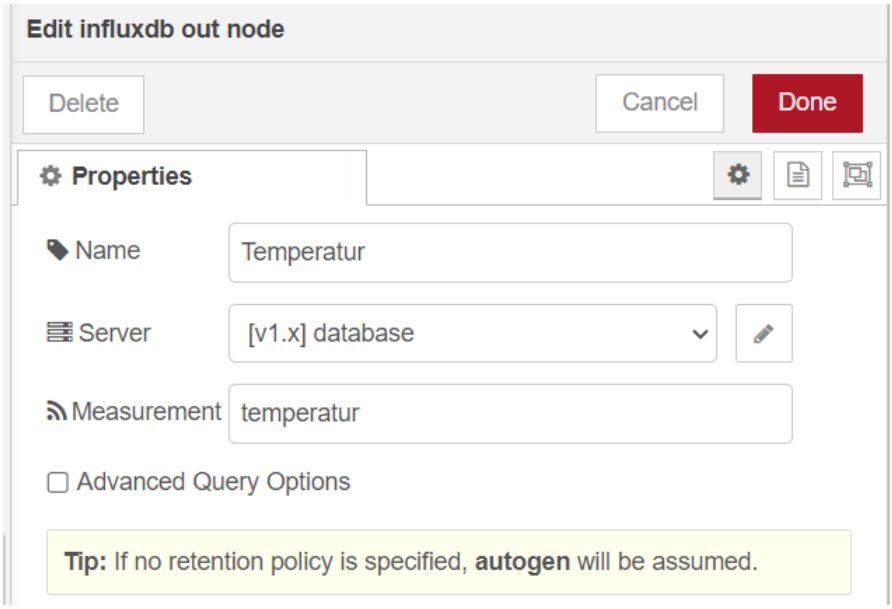
12) Trekk ut en influxdb out boks:



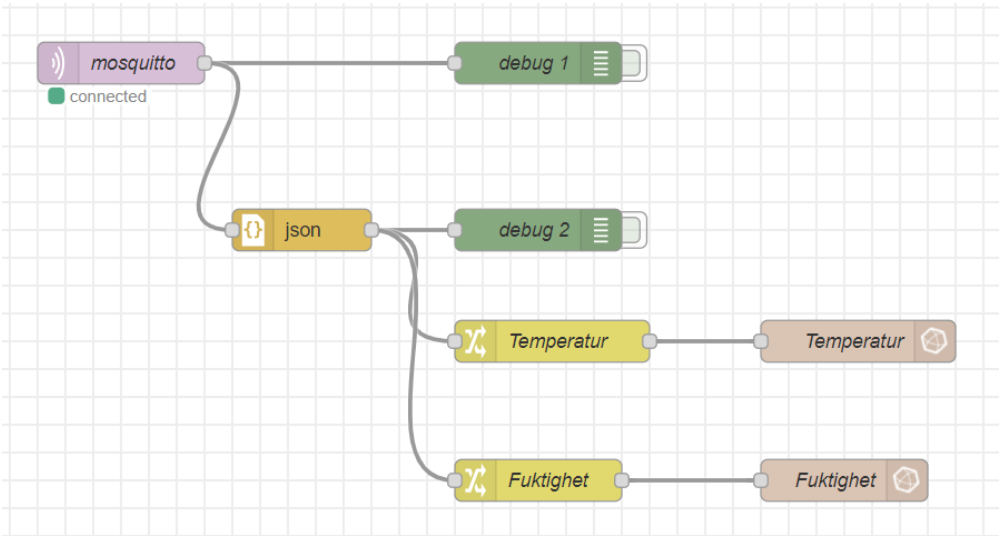
13) Konfigurer influxdb med blyanten og sett opp serveren med IP-en din og navnet på databasen:



14) Gi et navn til data du skal legge inn i databasen din. Dette navnet brukes i Grafana seinere.

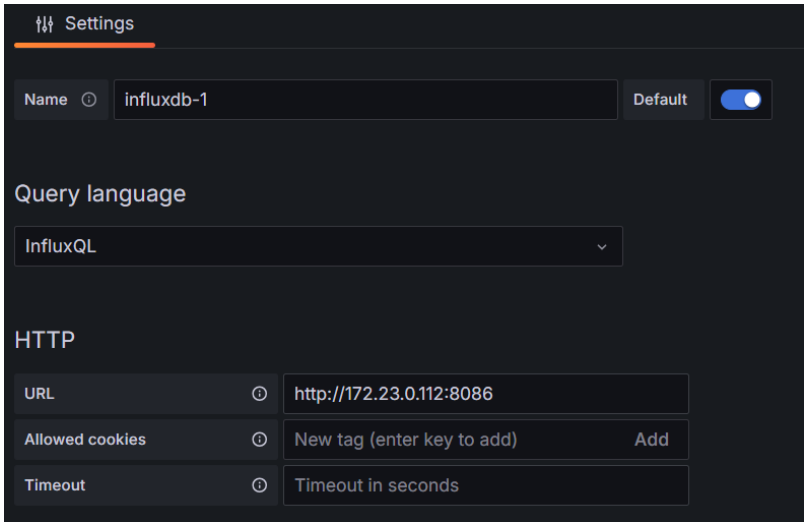


15) Kopier change blokk og database blokk for temperatur og lag nye blokker for fuktighet og Deploy til slutt.

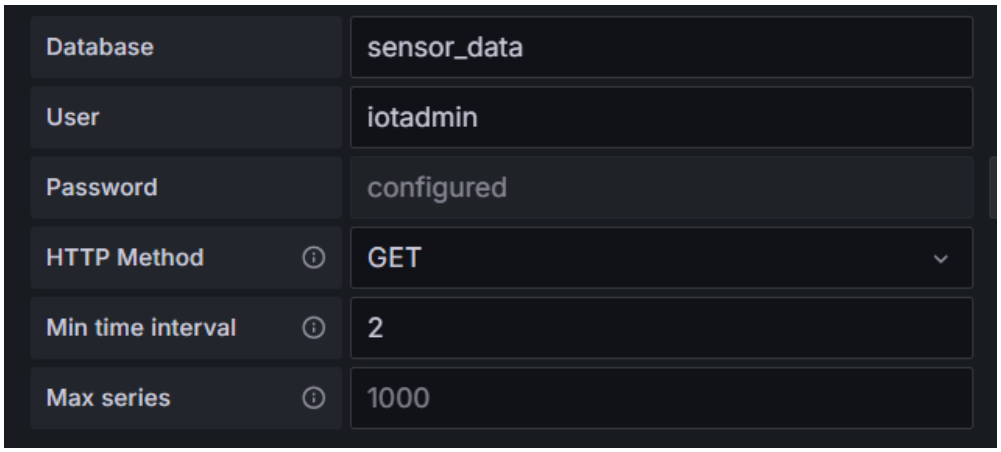


6 Visualiser dataene med Grafana

1) Gå inn på Grafana fra en nettleser på nettverket med http://<din-ip>:3000.
2) Logg på med admin/admin
3) Klikk på ‘Add your first data source’
4) Velg influxdb
5) Fyll inn URL: http://<din-ip>:8086

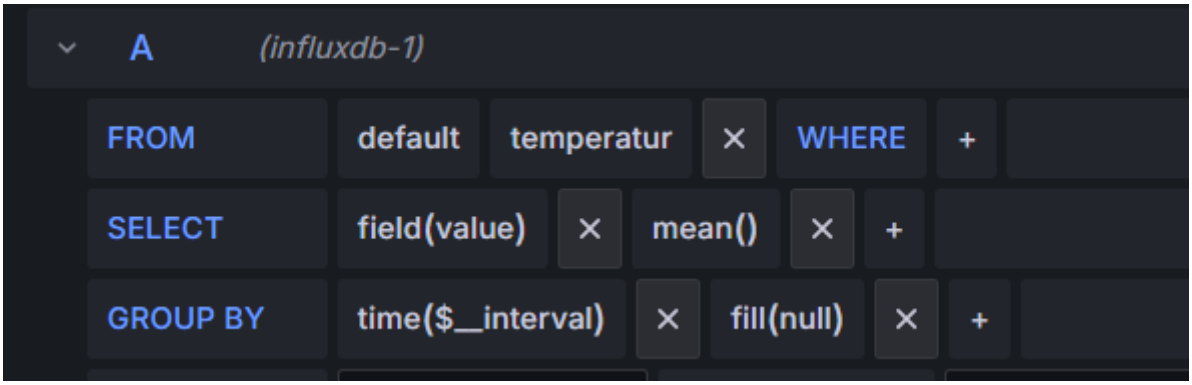


6) Fyll inn databasenavnet, GET, og 2:



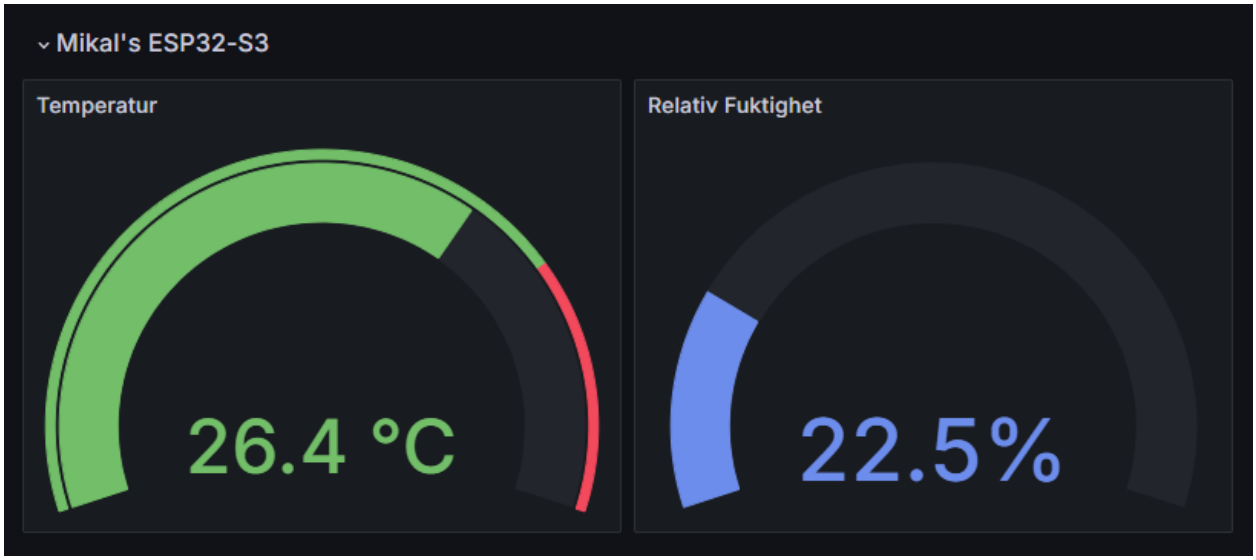
7) Save & test
8) Gå til Dashboard og begynn å lage et nytt ‘Panel’. Velg Add visualization

9) FROM er Measurment navnet brukt i influxdb i Node-RED.



10) Til høyre kan du konfigurere panelet og bruk «apply» for å vise endringer. Husk å «save» når du er ferdig.

11) Lag et nytt panel i samme dashboard for fuktighet. Eksempel vist nedenfor.



7 Forklaring:

Sensoren endrer motstand når temperaturen endrer seg og viser dette som en endret strøm eller spenning på mikrokontrolleren. **ESP32-S3 Mikrokontrolleren** oversetter strømmen eller spenningen den leser av på sensoren til hvilken temperatur det er. For å sende dataene videre trengs det en Protokoll – en avtale eller språk om hvordan kommunikasjonen skal være.

Mosquitto MQTT brokeren sitter mellom mikrokontrolleren og serveren for å håndtere kommunikasjonen.En server kan lese av data-ene som kommer inn via MQTT brokeren, men da leser vi jo akkurat bare hva som skjer akkurat nå. Hvordan vet vi "trenden" på temperaturen gjennom et døgn? Da må vi lagre dataene i en database.

Influx databasen er som et excell-ark. Her lagres data i skjemaer. I tillegg følger det med en masse kodemuligheter til å sette sammen dataene på utallige måter. Vi bruker Influx IOT-stacken men da dere satt opp LAMP-stack var det MySQL. SQL-varianter er nok de vanligste databasene. Ved å ha alle sensor data i databaser, så er de lagret over tid. Da kan vi feilsøke ved å se på trendene: kanskje lageret på en generator vibrerer ekstra mye når lageret er varmt? Da kan vi se sammenhengen mellom vibrasjoner og temperatur over tid for å analysere. Network Chuck på youtube har en fin innføring i det viktigste med databaser [4]

Det er tungvindt å gjøre queries — spørringer — på databaser. Det er langt bedre å ha grafiske fremstillinger som kakediagram, kurver og søyler for å vise hastigheter, temperaturer, vibrasjoner osv.

Grafana er et verktøy for å grafisk fremstille data. Den henter data "measurements" fra databaser ("excell-ark") og viser det som forskjellige klokker, søyler og kurver på "dashboards". Her viser en fyr Grafana[4] .

Node-Red er et "low code" programmeringspråk som syr alt sammen. Her programmerer vi ved å sette ut bokser som inneholder ferdig java kode som vi bare konfigurerer gjennom GUI-en. Det er ikke nødvendig å skrive kode slik vi gjorde med python på mikrokontrolleren. Her er en 5-åring som lager hjemmeside med Node-Red [5].

MING-stack, eller IOT-stack, er noe vi kan kalle en server som er satt opp for IOT og inneholder M for MQTT; I, for influxdb; N, for Node-RED; og G, for Grafana — alt man trenger for å ta i mot sensordata, lagre de i en database og sensor dataene over i et grafisk grensesnitt.

Docker sørger for at hver av delene i MING-stacken settes opp i konteinere. Konteinere har 3 fordeler: De kommer ferdig med det du trenger for å kjøre akkurat den tjenesten, du trenger ikke sette av RAM og CPU som ved virtualisering, og du vet at ingenting annet er installert der som kan tulle med programmet ditt. **Docker-compose** lar deg i tillegg automatisere oppsettet og **Portainer** lar deg administrere Docker-serveren din fra en nettleser på nettverket. Network Chuck har en fin forklaring[6].

LAMP-stack er en annen standard måte å sette opp serveren på: A er apache2 webserver, M er MySQL database, og P er PHP. Skulle vi brukt en LAMP til IOT ville vi brukt SQL database istedet for Influx og PHP i stedet for Node-RED. Apache-serveren kunne hostet en hjemmeside som presenterte dataene fra databasen, men LAMP er mer egnet for tradisjonelle webservere, mens MING er rettet mot IOT. Er du god på alle delene av en "stack" er du en "Full-stack dev".

7.1 Kommandoer for Docker:

Kommando	Beskrivelse	Eksempel
`docker ps`	Lister alle kjørende konteinere. `-a` viser alle beholderene	`docker ps -a`
`docker start`	Starter en eller flere stoppede kontainere	`docker start webserver`
`docker stop`	Stopper en kjørende konteiner	`docker stop webserver`
`docker exec`	Kjører kommandoer inne i konteinere	`docker exec -it webserver /bin/bash`
`docker images`	Lister alle Docker-imager på hosten.	`docker images`
`docker pull`	Henter et image eller et repository f	`docker pull ubuntu`
`docker build`	Bygger Docker-image fra en Dockerfile. Et image inneholder også konfigurasjonen som er gjort.	`docker build -t my-custom-image .`
`docker rmi`	Fjerner ett eller flere Docker-image.	`docker rmi my-custom-image`
`docker stats`	Viser statistikk over konteineren sin ressursbruk.	`docker stats`
`docker inspect`	Viser detaljer til Docker-objekter (beholdere, imager, nettverk, osv.). Nyttig for å få system- og konfigurasjonsdetaljer.	`docker inspect webserver`
`docker network`	Administrerer Docker-nettverk. Lar deg koble beholdere til nettverk, liste eksisterende nettverk, inspisere og fjerne nettverk.	`docker network create my-network`
`docker logs`	Henter loggene fra en konteiner. Dette er nyttig for feilsøking og overvåking av beholderens oppførsel.	`docker logs webserver`

7.2 Kommandoer for Influxdb:

Command	Description (Norwegian)	Example Use
`SHOW DATABASES`	Viser alle databaser i InfluxDB.	`SHOW DATABASES`
`CREATE DATABASE <name>`	Oppretter en ny database med det angitte navnet.	`CREATE DATABASE sensor_data`
`DROP DATABASE <name>`	Sletter den angitte databasen.	`DROP DATABASE sensor_data`
`USE <database>`	Velger en database for bruk i den påfølgende sesjonen.	`USE sensor_data`
`SHOW MEASUREMENTS`	Viser alle målinger i den gjeldende databasen.	`SHOW MEASUREMENTS`
`INSERT <measurement> ...`	Setter inn en datapunkt i en spesifikk måling.	`INSERT temp,location=north value=23.5`
`SELECT <field> FROM <measurement>`	Henter spesifiserte felter fra en gitt måling.	`SELECT value FROM temp`
`SHOW FIELD KEYS FROM <measurement>`	Viser alle felt-nøkler for en spesifikk måling.	`SHOW FIELD KEYS FROM temp`

8 Kilder

[1] «MING Stack for IoT • FlowFuse». Åpnet: 29. april 2024. [Online]. Tilgjengelig på: https://flowfuse.com/blog/2023/02/ming-blog/

[2] «Install Docker Engine on Ubuntu | Docker Docs». Åpnet: 29. april 2024. [Online]. Tilgjengelig på: https://docs.docker.com/engine/install/ubuntu/

[3] *SuperHouse #41: Datalogging with MQTT, Node-RED, InfluxDB, and Grafana*, (13. desember 2020). Åpnet: 29. april 2024. [Online Video]. Tilgjengelig på: https://www.youtube.com/watch?v=ffg3_1AgtyA

[4] *you need to learn SQL RIGHT NOW!! (SQL Tutorial for Beginners)*, (17. august 2022). Åpnet: 29. april 2024. [Online Video]. Tilgjengelig på: https://www.youtube.com/watch?v=xiUTqnI6xk8

[5] *How to make a website with Node-RED*, (20. desember 2017). Åpnet: 29. april 2024. [Online Video]. Tilgjengelig på: https://www.youtube.com/watch?v=SiTFHw8LQXc

[6] *you need to learn Docker RIGHT NOW!! // Docker Containers 101*, (3. april 2020). Åpnet: 29. april 2024. [Online Video]. Tilgjengelig på: https://www.youtube.com/watch?v=eGz9DS-aIeY

Filer:

(nås med wget i linux, eller bare hentes fra en nettleser)

Techne.guru/docker-compose.yml

Techne.guru/iot-server.pdf

Techne.guru/mosquitto.conf

Techne.guru/passwords.txt

Techne.guru/code.py

Techne.guru/code.py.zip