Opdrachten week 5 lab 2 – Wifi (IoT)

Vorig lab heb je gekeken hoe een verbinding kan worden opgezet door het gebruik van sockets. Dit lab gaan we verder kijken dan een lokaal netwerk en bekijken we een echte IoT (Internet of Things) toepassing die de basis zal vormen voor de eindopdracht. Aan het einde van dit lab, kun je vanaf het internet met behulp van een browser een ledje schakelen op je eigen LaunchPad en/of op de LaunchPad van een medestudent.

Je gaat leren:

- wat MQTT is;
- wat Node.js en Node-RED zijn;
- een integrale IoT oplossing te realiseren.

Eerst is een introductie tot MQTT benodigd. MQTT staat voor Message Queuing Telemetry Transport en is een protocol dat is bedacht om het voor IoT-apparaten gemakkelijker te maken data te versturen en te ontvangen[1]. Het is minder complex en vereist minder data dan HTTP. Wel heeft het ondersteuning voor beveiligde verbindingen via TLS[2] en is het ideaal voor het versturen van toepassingsspecifieke data. Figuur 1 laat zien wat de principes zijn van MQTT. MQTT heeft grofweg twee acties:

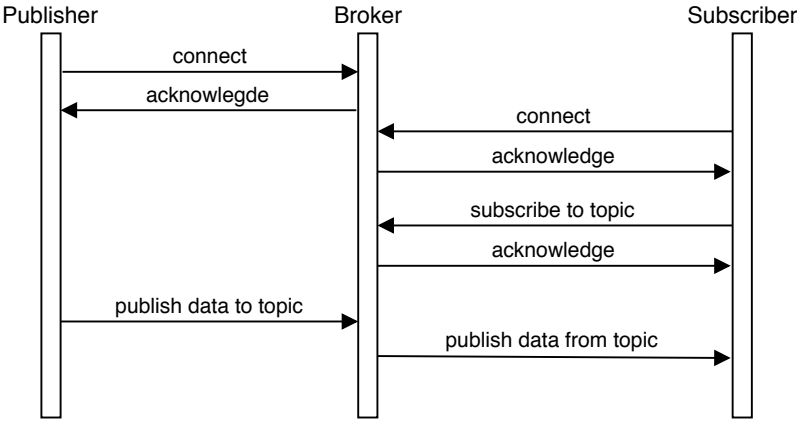
- subscribe (wordt lid van een topic);
- publish (maak een bericht beschikbaar op een topic).

Een topic is simpelweg een locatie in de database. Je kunt het vergelijken met een forum op het internet. Op het forum kunnen verschillende topics worden aangemaakt en in een topic kan informatie worden gepost.

De broker vormt de database van alle berichten die opgeslagen moeten worden en stelt deze database beschikbaar via het MQTT protocol. Voor de publisher voorziet het MQTT protocol in de mogelijkheid om te verbinden met een broker en data beschikbaar te maken (publish). Als subscriber kun je je aanmelden bij de broker en lid worden van een of meer topics, bijvoorbeeld `\sensoren\temperatuur\id123`. Wanneer de publisher iets nieuws stuurt, dan krijgen alle subscribers die lid zijn van dat topic bericht. Elke client van de broker kan één of beide rollen aannemen.

[1] Zie eventueel https://mqtt.org/.

[2] Transport Layer Security (TLS) zijn encryptie-protocollen die de communicatie tussen computers (bijvoorbeeld op het internet) beveiligen.



Figuur 1: Sequentiediagram waarin de basisbewerkingen van MQTT worden weergegeven. De tijdvolgorde loopt van boven naar beneden.

Gelukkig heeft Texas Instruments een MQTT bibliotheek beschikbaar gesteld zodat we weinig hoeven te doen om het toe te passen.

5.2.1

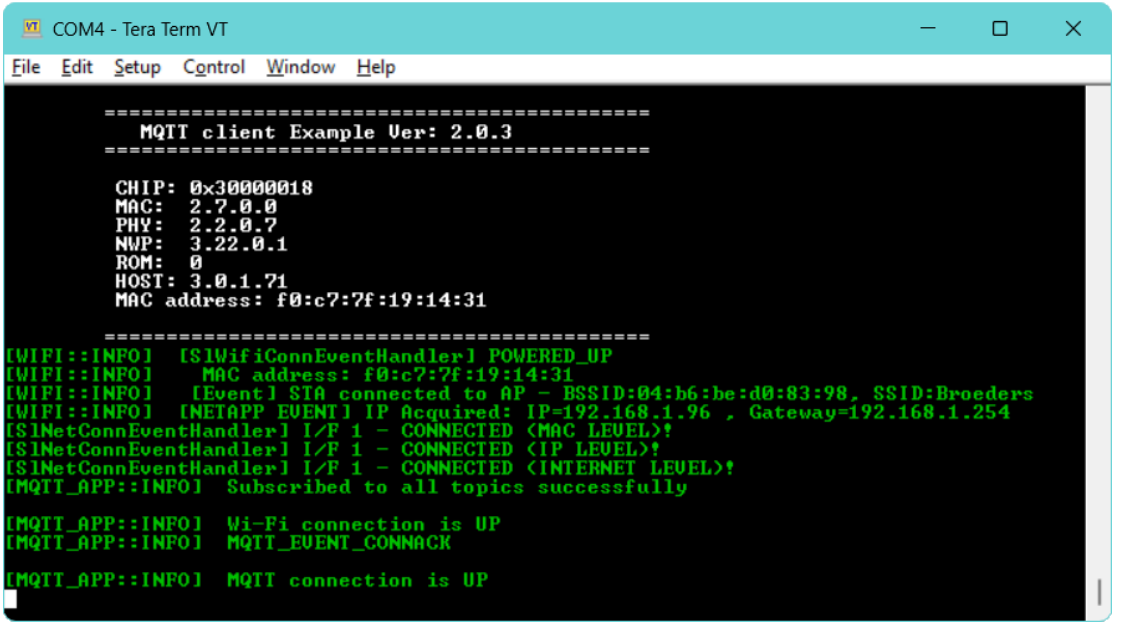
Open CCS en importeer het project `mqtt_client_CC3220S_LAUNCHXL_tirtos_ccs` vanuit de SDK.

- Maak de Debug configuratie Active en delete vervolgens de MCU+Image configuratie.
- Verwijder `image.syscfg` en `MCU+Image` uit het project.
- Maak in `wifi_settings.h` de volgende aanpassingen:
  - maak `AP_SSID` op regel 101 gelijk aan de SSID (naam) van jouw WLAN (WiFi); [3] [4]
  - maak `AP_PASSWORD` op regel 102 gelijk aan het wachtwoord van jouw WLAN.
- Verander op regel 136 het `ClientID` in je studentnummer.

[3] Thuis gebruik je uiteraard je eigen wifi-netwerk. Op school maak je gebruik van een wifi-hotspot op een telefoon (zoek eventueel op internet uit hoe dat moet op jouw telefoon) of van de EMS20 routers. We maken geen gebruik van het wifi-netwerk van de hogeschool (eduroam) omdat je dan je gebruikersnaam en wachtwoord in je code zou moeten zetten en dat willen we om veiligheidsredenen vermijden.

[4] Als je een iPhone gebruikt, dan heeft de standaard hotspot-SSID een omgekeerde apastrof in de naam, bijvoorbeeld “Daniel‘s iPhone”. Als `AP_SSID` moet je dan `"Daniel\xE2\x80\x99s iPhone"` invullen in plaats van `"Daniel's iPhone"`.

Kijk of het project compileert, open een terminal, stel de baudrate in op `115200` en debug het project. Als alles goed gaat ziet de uitvoer er uit zoals in figuur 2.

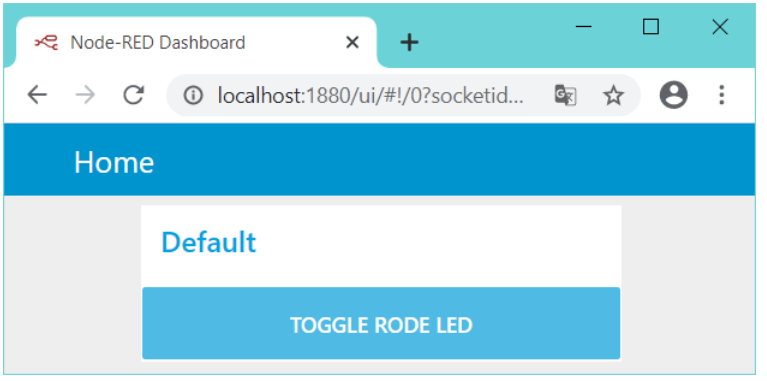


Figuur 2: Uitvoer wanneer succesvol met een wifi-netwerk en de broker is verbonden.

5.2.2 De topics waarmee het standaard project verbindt (als subscriber) staan vast in de code. Pas de code in `mqtt_client_app.c`, regel 840 tot en met 842, aan zodat het programma lid wordt van de topics `ems20/<jouw studnr>/led<x>` in plaats van `cc32xx/ToggleLED<x>`. Voer het project opnieuw uit en laat het aan staan. Nu moeten we er nog voor zorgen dat we via een browser een bericht kunnen sturen naar de topic zodat de LaunchPad hier bericht van krijgt en de betreffende led kan schakelen.

Er zijn diverse manieren om vanaf een pc verbinding te maken met een MQTT broker. Wij kiezen om dit te doen via een javascript client met ingebouwde webserver, zodat het via de browser is te benaderen. We gaan gebruik maken van Node.js om hierop een server te draaien genaamd Node-RED[5]. De Node-RED server stelt ons in staat om heel gemakkelijk een knopje te koppelen aan de MQTT-topic zoals weergeven in figuur 3.

[5] Zie eventueel `https://nodered.org/`.



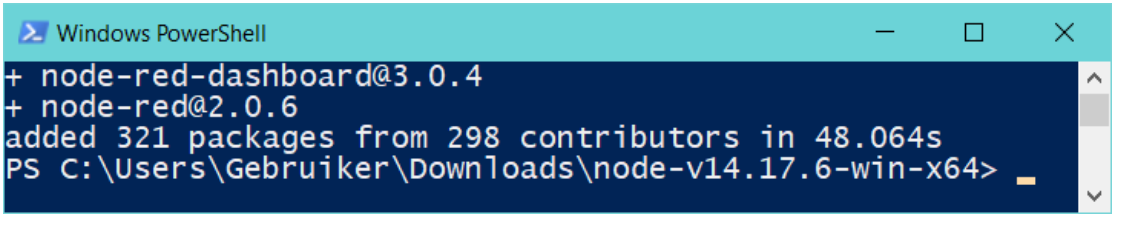
Figuur 3: Voorbeeld van de uiteindelijke webpagina.

5.2.3 Eerst moeten Node.js, Node-RED en Node-RED-dashboard geïnstalleerd worden.

- Download de Windows Installer (msi) van https://nodejs.org/en/download/ en installeer Node.js.
- Open Windows PowerShell
- In het venster wat opent plak je het onderstaande commando en druk je op enter om het uit te voeren.

  ```
  npm install -g --unsafe-perm node-red node-red-dashboard
  ```

  Dit commando gebruikt de Node.js package manager om de packages node-red en node-red-dashboard te installeren. Er verschijnen een aantal *warnings* maar die kun je negeren. Als het klopt zie je uiteindelijk hetzelfde als in figuur 4 wanneer de installatie klaar is.



Figuur 4: Uiteindelijke uitvoer van het commando.

Het is goed mogelijk dat je op dit moment de volgende foutmelding krijgt: `.\node-red : File node-red.ps1 cannot be loaded because running scripts is disabled on this system`. Dit kun je oplossen door het volgende commando in de PowerShell te gebruiken:

```
Set-ExecutionPolicy -Scope CurrentUser RemoteSigned
```

Vul `Y` in als de prompt vraagt of je de execution policy wilt aanpassen.

5.2.4 Nu alle benodigde Node-RED software is geïnstalleerd kunnen we het gaan gebruiken! Type in dezelfde PowerShell `node-red` om het te starten.

Zoals te zien in de uitvoer, heeft het een server gestart op poort 1880 en IP-adres 127.0.0.1[6].

5.2.5 Open een browser en ga naar `http://localhost:1880/`. Als het klopt zie je een scherm vergelijkbaar met figuur 5.

5.2.6 In het linkerdeel van Node-RED, zie figuur 5, zie je een kopje common met daaronder blokken die je kunt slepen naar het middendeel. Deze blokken zijn bouwstenen waarmee je een web-applicatie kunt bouwen en deze bouwstenen worden *nodes* genoemd. Naast de categorie common zijn er, onder andere, ook de categorieën function, network en dashboard.

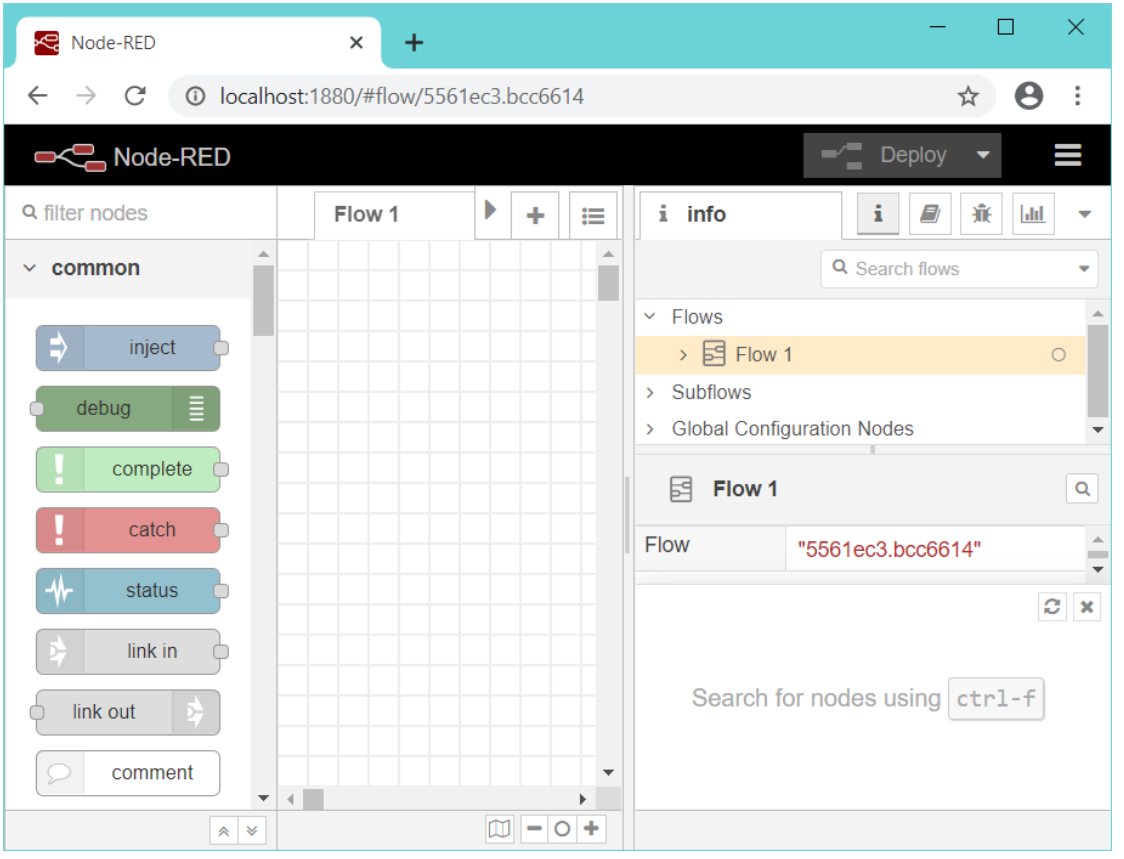
Sleep vanuit de categorie network de node *mqtt out* en vanuit de categorie dashboard de node *button*. Verbind de rechterkant van de *button*-node met de linkerkant van de *mqtt*-node.

- Dubbelklik op de *button*-node. Geef de knop een goed *Label* zodat je dadelijk de knop in het dashboard kunt herkennen. Bij *Payload*[7] kun je `1` invullen.

  Maakt een nieuwe *Group* aan (voor de grafische interface) door op het potlood te klikken (rechts naast "Add new ui_group..."). Geef de groep een naam. Maak tevens een nieuwe *tab* aan voor in de groep. Druk achtereenvolgens op Add > Add > Done.
- Nu de knop is ingesteld, moet het *mqtt*-node nog ingesteld worden. Dubbelklik op de *mqtt*-node. Stel bij *Topic* één van jouw topics in: `ems20/<jouw studnr>/led1`. Voeg een server toe door op het potlood te klikken. De server die het project gebruikt is `broker.hivemq.com` op poort 1883. Druk achtereenvolgens op Update > Done.

[6] Dit speciale IP is een loopback naar localhost. Dit adres verwijst altijd naar de computer zelf.

[7] De payload is het bericht dat wordt verstuurd als op de button wordt geklikt. Omdat we in dit geval alleen willen weten dat er op de button is geklikt maakt het niet uit welke payload we kiezen.



Figuur 5: Node-RED

5.2.7 Nu alles is geconfigureerd kan op Deploy worden geklikt. Het MQTT blok zou *connected* moeten weergeven.

Ga nu naar `http://localhost:1880/ui`. Als het klopt is dit vergelijkbaar met figuur 3.

Als alles goed is gegaan, kun je op de knop drukken en schakelt jouw rode led.

Wat gebeurt er nu eigenlijk? De *mqtt*-node, is in de achtergrond gewoon een functie die verbinding maakt met de ingestelde broker `broker.hivemq.com`. De *button*-node wordt in de dashboard weergeven als een aanklikbare knop.

Op het moment dat op de knop wordt geklikt zal de *button* zijn *payload* (1) *publishen* op het topic dat in de *mqtt*-node is ingesteld.

Jouw LaunchPad had zich aangemeld voor ditzelfde topic en zal dus bericht krijgen met de payload (1). Afhankelijk van de ingestelde topic zal de LaunchPad dan een van de leds schakelen.

5.2.8 Herhaal het proces nog een keer om nog twee knoppen toe te voegen voor jouw andere 2 leds en topics.

Het is natuurlijk leuker om de led te kunnen schakelen van een andere LaunchPad, die niet is verbonden met jouw computer. Vind een medestudent, vraag zijn topics en stel dit in bij Node-RED. Deploy opnieuw, ga naar het dashboard en druk op de knoppen. De leds op de LaunchPad van je medestudent zouden nu moeten schakelen!

5.2.9 Het is natuurlijk ook mogelijk dat meerdere LaunchPads zich bij hetzelfde topic aanmelden. Dan kan één iemand de leds van meerdere LaunchPads schakelen. Je kunt dit uitproberen.

5.2.10 Je hebt nu drie topics gebruikt. Voor elke led heb je één aparte topic gebruikt. Het is ook mogelijk om slechts één topic te gebruiken met verschillende payloads. Het nummer in de payload bepaald dan welke led geschakeld moet worden. Pas de programma's aan zodat nog maar met één topic wordt gewerkt.

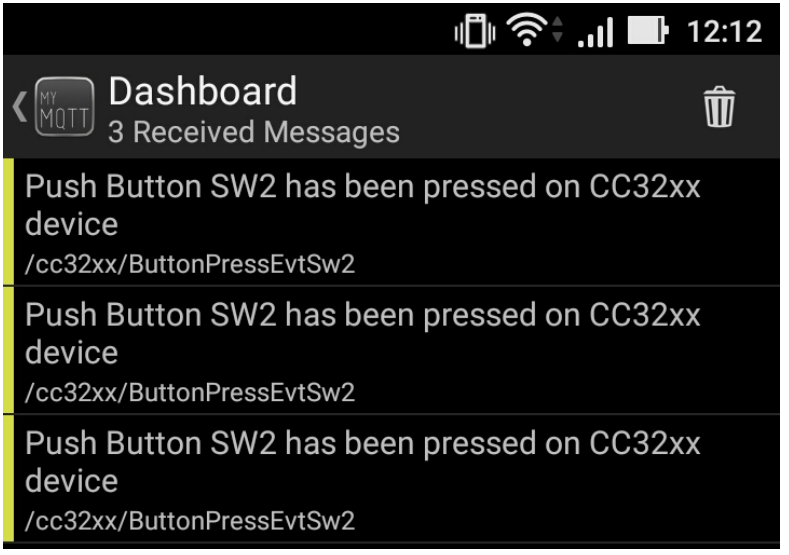
5.2.11 Het programma op je LaunchPad zal een topic publiceren als op SW2 (het knopje aan de linkerkant) wordt gedrukt. Geef de topic een specifieke naam en pas de data die verstuurd wordt naar wens aan. Breid de Node-RED applicatie nu uit zodat deze abonneert op dit topic en het ontvangen bericht voorleest in de browser. Maak gebruik van de *mqtt in*-node en de *audio out*-node.

5.2.12 Je kunt ook een standaard MQTT-client gebruiken om te communiceren met je LaunchPad. Op jouw computer kun je bijvoorbeeld MQTTX[8] gebruiken. Op jouw Andoid telefoon kun je bijvoorbeeld MyMQTT[9] gebruiken, zie figuur 6.

Het idee achter MQTT is nu duidelijker. Als een IoT-sensor een nieuwe waarde heeft, kan hij die publishen naar een MQTT-server. Nu kunnen alle clients die lid zijn van dat topic,

[8] Zie https://mqttx.app/downloads.

[9] Zie https://play.google.com/store/apps/details?id=at.tripwire.mqtt.client&hl=en.



Figuur 6: Berichten van de LaunchPad weergegeven op een telefoon met behulp van MyMQTT.

meteen de waarde ontvangen zolang de client maar een internet verbinding heeft! Andersom kunnen 100 sensoren naar één topic data sturen, zodat één client alle data binnen kan halen.

MQTT is een mogelijke manier om data naar de cloud te versturen en te benaderen, maar natuurlijk niet de enige. Als je later voor stage of afstuderen een vergelijkbare opdracht krijgt, kun je op dat moment kijken wat de beste optie is! De softwarewereld verandert snel.