Haaga-Helia
ammattikorkeakoulu Oy

Node-RED:in käyttö Raspberry PI tietokoneessa ja ohjelmien toteuttaminen MQTT-protokollaa hyödyntäen

Marco Brandt

Opinnäytetyö
Tietojenkäsittelyn
koulutusohjelma
2021

Haaga-Helia
ammattikorkeakoulu Oy

Tekijä(t)
Marco Brandt

Koulutusohjelma
Tietojenkäsittelyn koulutusohjelma

Raportin/Opinnäytetyön nimi
Node-RED:in käyttö Raspberry PI tietokoneessa

Sivu- ja liitesivumäärä
43

Opinnäytetyössäni asennan Node-RED ohjelmointi sovelluksen Raspberry Pi tietokoneelle ja toteutan useamman sovelluksen kyseisellä tietokoneella.

Raspberry on Raspberry Pi Foundationin omistama mikrotietokone, joka toimii yhdeltä pienikokoiselta piirikortilta. Raspberry Pi on edennyt jo neljänteen sukupolveen ja sisältääkin nykyään melko tehokkaan prosessorin, jonka tehot riittävät Linux käyttöjärjestelmän asentamiseen. Raspberry OS on oletus käyttöjärjestelmä laitteella ja sen käyttöönottaminen on helppoa. Muutamia tietoturvaan liittyviä asetuksia voi ottaa käyttöön järjestelmässä, jotka vaativat käyttäjältä kokemusta Linuxin komentorivi komennoista.

Node-RED on graafinen ohjelmointityökalu, joka perustuu Node.js ajoympäristöön. Node-RED käyttää JavaScript ohjelmointikieltä toiminnoissaan, mutta ohjelmointi tapahtuu vetämällä Node-RED työpöydälle ohjelmointipalikoita. JavaScript tulee käyttöön, kun tehdään monimutkaisempia ohjelmia. Pienen sovelluksen tekeminen, jossa seurataan Raspberry Pi tietokoneen prosessorin lämpötilaa, ei vaadi kuin muutaman rivin ohjelmointia ja saamme tulosteen arvoista.

MQTT-viestintäprotokolla on alun perin kehitetty kahden laitteen väliseksi kommunikaatioksi, jonka tavoitteena oli tehdä mahdollisimman tehokas keino kommunikoida pienen sähköenergian ja datakaistan avulla. Keksijä alun perin kehitti teknologian satelliittien ja öljyputkien väliseen kommunikaatioon, jolloin sähkön- ja tiedon määrä vaikuttivat merkittävästi kustannustehokkuuteen.

Node-RED hyödyntää MQTT-protokollaa erittäin helposti, koska siinä on valmiiksi asennettuna MQTT-työkalut. Sillä voi vastaanottaa Helsingin Seudun Liikenteen MQTT-palvelimen tietoja ja käyttää niitä haluamallaan tavalla. Tiedon määrä on valtava, jolloin tarvitsemme kehittyneempiä funktiota niiden karsimiseen. Node-RED:llä voidaan ottaa junien tai bussien liikennetiedot ja sijoittaa ne kartalle tarkasteltavaksi.

Node-RED voi olla jatkuvassa käytössä ja aina saavutettavissa Raspberry Pi mikrotietokoneen avulla. Laitteen pieni koko ja energiatehokkuus tekee siitä loistavan IoT-laitteen. Node-RED on myös kevyt sovellus, joka ei vaadi kirjautumista Rasperryyn. Node-RED toimii verkkoselaimen kautta ja sitä voidaan käyttää verkon kautta laittamalla kyseisen tietokoneen IP-osoitteen selaimeensa. Tämä on huomioitava tietoturvassa.

Asiasanat
Tietokoneohjelmointi, esineiden internet, Raspberry Pi, mikrotietokoneet

Sisällys

1 Johdanto 1
2 Asioiden Internet 3
2.1 Vuopohjainen ohjelmointi 3
2.2 Node-RED 5
2.2.1 Node.js ja Npm 5
2.2.2 Node-RED ohjelmointi 6
2.3 MQTT 7
2.4 JavaScript 8
2.5 Raspberry Pi 10
2.5.1 Raspberry Pi 4 model B ominaisuudet 11
2.5.2 Raspberryn käyttöjärjestelmät 11
2.5.3 Etäyhteydet Raspberryllä 12
2.5.4 SSH-yhteys 13
2.5.5 Etätyöpöytä 14
3 Node-RED-sovelluksen käyttöönotto Raspberry Pi tietokoneessa 15
3.1 Johdanto 15
3.2 Raspberryn käyttöönotto 16
3.3 Raspberryn ohjelmien asentaminen ja LXTerminal 17
3.3.1 Raspberry OS päivittäminen 18
3.4 Raspberryn turvallisuus 19
3.4.1 Uusi tunnus 20
3.4.2 Palomuuri 21
3.5 Node-RED asentaminen 22
3.5.1 Node-RED HTTPS-suojaus 23
3.5.2 Node-RED vuoeditorille salasana 23
3.6 Node-RED:in käyttö Raspberry PI tietokoneessa 25
3.6.1 Node-RED vuoeditori 25
3.7 Ensimmäinen ohjelma Raspberrylle 26
3.8 Ensimmäisen ohjelman kehittäminen pidemmälle 28
3.9 MQTT Node-RED sovellus 32
3.9.1 HSL MQTT-broker 32
3.9.2 HSL MQTT hyötykuorma 35
3.9.3 Node-RED hyödyntämässä HSL MQTT-palvelua 36
4 Pohdinta 42
LÄHTEET 44

1 Johdanto

Opinnäytetyöni koskee Node-RED ohjelmointi ympäristöä ja sen käyttöä Raspberry Pi tietokoneella. Ideana on tehdä pienikokoinen tietokone, jossa on Node-RED sovellus ja on aina käytettävissä.

Tässä opinnäytetyössä käymme ensin läpi Raspberryn käyttöönoton alusta asti, jolloin aiempaa osaamista ei vaadita. Tutustumme eri vaihtoehtoihin Raspberryn käytössä ja miten vaihtoehdot vaikuttavat sen toimintaan. Käyttöjärjestelmä on yksinkertainen Raspberry OS ja opastan lyhyesti sen käyttämistä. Teemme perusasennuksen ja teemme tietokoneesta tietoturvallisen, jolloin sen voi jättää huoletta päälle jatkuvaankin käyttöön.

Kun olemme saaneet laitteen käyttökuntoon voimme viimein aloittaa ohjelmoinnin Node-RED sovelluksella. Node-RED on Node.js ajoympäristössä toimiva ohjelmointityökalu. Ohjelmointi tapahtuu verkkoselaimen kautta graafisen käyttöliittymän avulla. Node-RED ohjelmointi nimensä mukaisesti tarkoittaa ohjelmointia node-moduuleilla. Nodet vedetään sivustolla olevasta paletista ja yhdistetään toisiinsa viivoilla tai langoilla, jolloin ne toimivat kuten funktiot ja metodit perinteisessä ohjelmoinnissa. Node-RED visuaalisella ulkonäöllään tekee ohjelmoinnin luettavaksi myös henkilöille, joilla ei välttämättä ole kokemusta ohjelmoinnista.

Node-RED:in helppous on myös sen heikkous, koska se on helppo ottaa käyttöön ja se helposti myös pysyy päällä, vaikka ei sitä käyttäisi. Kyseessä on kuitenkin täysin varteenotettava ohjelmointi ympäristö. Node-RED:in turvaamme ensimmäisenä, jolloin ei tarvitse huolehtia sen toiminnasta kesken ohjelmoinnin.

Käytännössä käyn läpi eri moduuleja ja niiden toimintaa. Teen muutamia esimerkkejä, miten ohjelmointia voidaan toteuttaa ja yhden laajemman projektin missä käytän MQTT-protokollaa ja seuraan HSL liikennettä reaaliajassa. Esittelen myös verkkosivulla käytettävät dashboard ja worldmap laajennukset, joilla voimme luoda näyttäviä sivustoja.

Avainsanoja

Back-end – Palvelinpuolen laitteet ja ohjelmat
Bcrypt – Tehokas salausalgoritmi.
Etätyöpöytä – Tietokoneen työpöytä käytettynä verkon kautta.
Front-end – Käyttäjälle näkyvät palvelut ja ohjelmat.
IoT – Laitteiden internet. Osa jääkaapeista ja pesukoneista ovat nykyään jatkuvassa yhteydessä internettiin.
JavaScript – Node-RED käyttää JavaScript-ohjelmointikieltä eri toiminnoissaan.
Linux – Toiseksi suosituin käyttöjärjestelmä, useita eri jakeluita.
MQTT – Kevyt kommunikointi protokolla eri laitteiden välille.
Node – Ohjelmointipala Node-RED vuoeditorissa.
Node.js – Ajoympäristö, joka mahdollistaa JavaScriptin ajamisen back-endissä.
Node-RED – Ohjelmointi työkalu, jolla voidaan luoda mm. yhteyksiä eri verkkopalveluiden väliin.
OSX – Applen oma käyttöjärjestelmä.
Palomuuri – Käytetään verkosta tulevien hyökkäyksien ja urkintayritysten estämiseen.
Raspberry OS
Raspberry PI – Pieni tietokone, jota voidaan käyttää työasemana tai palvelimena.
SSH – Turvallinen kommunikointi tapa tietokoneiden välille.
Vuopohjainen ohjelmointi – IBM:n kehittämä ohjelmointi tapa.
Windows – Microsoftin oma käyttöjärjestelmä. Yleisin käyttöjärjestelmä.
Xdrp – Linux pohjainen etätyöpöytä sovellus.

2 Asioiden Internet

IOT (Internet of Things) eli asioiden internet termiä käytetään laitteista, jotka toimivat internetissä itsenäisesti ja keräävät dataa ympäristöstään. IOT-laitteena voi olla esimerkiksi jääkaappi, joka tarkkailee jääkaapin sisältöä ja lämpötilaa tai kuntoranneke, joka tarkkailee käyttäjänsä sykettä ja liikuntatottumuksia. IOT-laitteita kehitetään jatkuvasti ja niiden lukumäärä oli arviolta 31 miljardia kappaletta vuonna 2020 (Lynkova, Darina 2019).

Tavalliselle käyttäjälle IoT-laitteiden toiminta voi kuitenkin vielä olla vaikeasti ymmärrettävää. Yksinkertaisen IoT-laitteen tekeminen kotona ei kuitenkaan vaadi kuin hieman vaivaa ja aikaa. Yleensä tämän kaltaisilla laitteilla pyritään automatisoimaan jotain mitä ei itse jaksa seurata, kuten ulkovalojen laittaminen päälle auringon laskiessa tai kylmälaitteiden lämpötilan tarkkailua. Teollisuudessa voidaan monitoroida eri koneiden toimintaa IoT-tietokoneella, joka raportoi poikkeamista.

Raspberry tietokoneen ja Node-RED sovelluksen avulla voidaan tehdä edullinen tietokone, joka esimerkiksi seuraa säätietoja, maanjäristyksiä tai vaikka osakemarkkinoita. Seurattua dataa voidaan tallentaa, lähettää eteenpäin tai luoda hälytyksiä, jos laitetut rajaarvot ylittyvät. Node-RED on helposti opittava ohjelmointi ympäristö, jonka avulla voi luoda interaktiivisemman kotiympäristön.

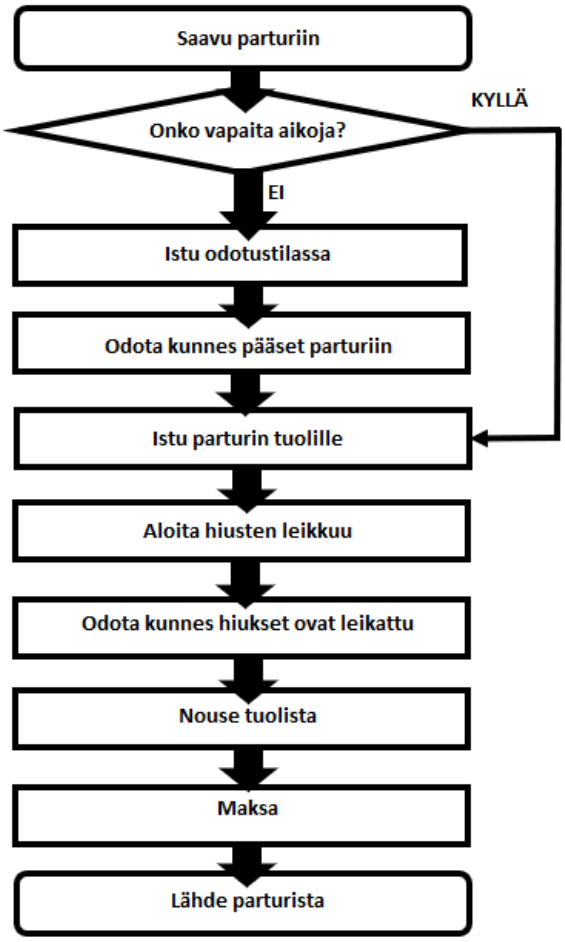
Vaikka Node-RED on helppo käyttää, vaati sen käyttäminen perehtymistä ja onkin hyvä tuntea esimerkiksi mitä vuopohjainen ohjelmointi tarkoittaa, mikä on Raspberry Pi ja miten Node-RED toimii. Näiden lisäksi on myös hyvä tuntea laitteiden turvallisuus, koska IoT-laitteet internet-verkkoon yhdistettynä voivat olla kohde hakkereille.

2.1 Vuopohjainen ohjelmointi

Flow-pohjainen ohjelmointi voidaan suomentaa vuopohjaiseksi ohjelmoinniksi. Vuopohjainen ohjelmointi on ollut käytössä jo pitkään IBM:llä ja sillä on pitkä historia.

Vuopohjaisen ohjelmoinnin kehitti 1970-luvulla IBM:llä työskennellyt J. Paul Morrison. Vuopohjaiseen ohjelmoinnin syntyyn vaikutti IBM:n käytössä olleet simulointikielet, kuten GPSS (General Purpose Simulation System) (Morrison, J Paul s.a.). GPSS koostuu palikoista, joiden avulla voidaan mallintaa haluttu simulaatio visuaaliseksi malliksi palikkadiagrammin avulla. Palikkadiagrammin pohjalta voidaan helposti ohjelmoida simulaatio tietokoneelle. (Thesen 1978 s. 238.)

Yksinkertainen esimerkki palikkadiagrammilla toteutetusta simulaatiosta on esimerkiksi parturissa käynti. Menet parturiin ja kyselet vapaata parturiaikaa, jos vapaata siirry kohtaan x, jos ei ole vapaata aikaa siirry odottamaan odotustilaan ja odota kunnes pääset kohtaan x. Palikkadiagrammi tälle simuloidulle tapahtumalle löytyy kohteesta kuva 1.



Kuva 1. GPSS flow mallinnus parturissa käyntiin (Thesen 1978, s. 216).

IBM otti vuopohjaisen ohjelmoinnin käyttöön omissa projekteissaan 70-luvulla, mutta vasta vuonna 1994 antoi sille nimen ”Flow-Based-Programming” (FBP). J. Paul Morrison kertoo erään kanadalaisen pankin edelleen käyttävän tuotantokäytössä hänen yli 40 vuotta sitten kehittämää teknologiaa. FBP pohjalta kehitetty teknologia on tietenkin jatkanut kehittymistään aina tähän päivään asti. (Morrison, J Paul s.a.)

2.2 Node-RED

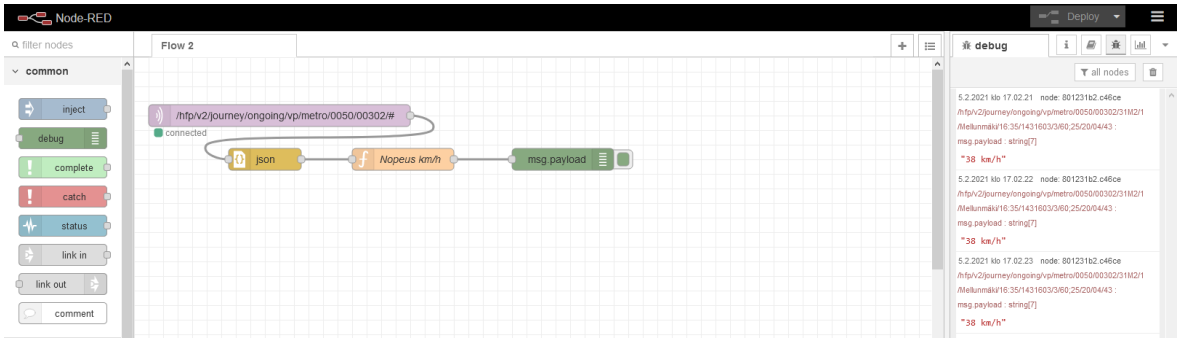
Node-RED on ohjelmointityökalu, jolla voi toteuttaa vuopohjaista ohjelmointia. Node-RED ottaa FBP (Flow Based Programming) laatikot ja antaa niille omat toiminnot, jolloin vuopohjainen ohjelmointi tulee mahdolliseksi uudella tavalla. Node-RED nimeää laatikoita nodeiksi ja työtilaan laitetut nodet yksinkertaistavat muuten monimutkaista toimintaa. Aloittelijakin voi hahmottaa mitä ohjelmassa tapahtuu, vaikka ei ymmärtäisi jokaista yksittäistä koodin pätkää nodejen sisällä. (OpenJS Foundation 2021a.)

Ohjelmointikielenä Node-RED syntyi Nick O'Learyn ja Dave Conway-Jonesin sivuprojektin tuloksena heidän työskennellessään IBM:llä. Node-RED eteni toteuttamiskelpoisuutta esittelevästä teknologiasta työkaluksi, joka pystyy laajentumaan mihin suuntaan tahansa. Node-RED on ollut avointa lähdekoodia vuodesta 2013. (OpenJS Foundation 2021a.)

2.2.1 Node.js ja Npm

Node.js on Javascript pohjainen ajoympäristö, jonka avulla voidaan luoda erilaisia verkkosovelluksia (OpenJS Foundation 2021b). Yksinkertaisimmillaan muutamalla rivillä koodia voi saada verkkoon "Hello World"-tyylisen verkkosivun. Monimutkaisimmillaan se on ollut tai on edelleen käytössä suurissa yrityksissä, kuten Microsoft (Baxter-Reynolds, Matthew 2011), Netflix (Xiao, Yunong 2014) ja Paypal (Scott, Bill 2013). Node.js mahdollisti JavaScriptin käytön myös verkkosivujen ulkopuolella eli pystytään toteuttamaan back-endin JavaScriptillä (Flanagan 2020). Back-end tarkoittaa ohjelmoinnissa palvelinpuolen ohjelmointia eli miten asiat toimivat ja front-end on mitä käyttäjä tai asiakas näkee käyttäessään sovellusta tai verkkosivua.

Node-RED käyttää Node.js ajoympäristöä luodakseen vuoeditorin verkkoselaimelle ja sovellusta käytetäänkin pääasiassa verkkoselaimen kautta. Käyttöliittymä (Kuva 2) on aluksi melko yksinkertaisen oloinen, mutta moduuleihin syventyminen mahdollistaa monimutkaisemman ohjelmoinnin.



Kuva 2. Node-RED työtilassa toteutettu yksinkertainen MQTT-ohjelma.

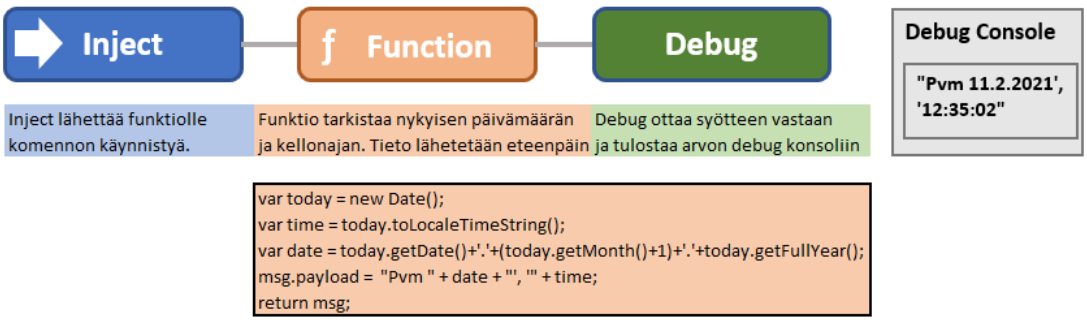
Lisäksi vaaditaan myös npm ohjelmistorekisteri. Npm on maailman suurin ohjelmistorekisteri, josta Node-RED ladataan tietokoneelle. Npm on hyvin yleinen työkalu ohjelmoinnissa, koska se mahdollistaa valmiiden sovellusten ja ohjelmakoodien lataamisen yksinkertaisilla komennoilla. (Npm 2021.) Npm komentoja käytettäessä tulee olla hyvin tarkka mitä kirjoittaa, koska kirjoitusvirheisiin pohjautuvat haitalliset ohjelmat voivat olla tietoturvariski (Spring, Tom 2017). Yksinkertainen kirjoitusvirhe voi asentaa koneelle haittaohjelman, jonka toimintaa ei välttämättä edes huomaa, kunnes myöhemmin tajuaa, ettei asennettu ohjelma toteuta haluttua toimintoa.

2.2.2 Node-RED ohjelmointi

Node-RED asennuksen jälkeen voidaan aloittaa työskentely käyttöympäristössä. Node-RED toimii kaikilla yleisimmillä verkkoselaimilla, kuten Safari, Firefox, Chrome ja Edge. Työtilan muutokset tallentuvat vasta kun painat ”Deploy”, jolloin ohjelma ottaa kyseisen työtilan käyttöön.

Ohjelmointi Node-RED sovelluksella tapahtuu vetämällä paletista node-moduuleja työtilaan. Esimerkiksi laittamalla noden vastaanottamaan dataa voi ohjelman hakea tietoa verkosta, tietokoneelta tai ulkoisesta laitteesta. Tämän jälkeen voit käsitellä haluamallasi tavalla syötettyä tai vastaanotettua tietoa ja tulostaa sen Node-RED konsoliin, verkkosivulle tai muulle alustalle halutessasi. Node-RED pystyy lähettämään tiedot eteenpäin MQTT tai JSON muodossa, jolloin kommunikaatio muiden laitteiden ja ympäristöjen kanssa on helppo toteuttaa.

Kuva 3 esimerkissä käytetään inject-nodea käynnistämään funktio-node, johon on Javascriptillä ohjelmoitu komento tarkistamaan nykyinen päivämäärä ja kellonaika. Funktio tämän jälkeen lähettää tiedon debug-nodeen, joka tulostaa sen Node-RED konsoliin selaimessa. Debug-nodella on helppo tarkistaa tiedon kulkeutuminen nodejen välillä.



Kuva 3. Node-RED node rakenne.

Sovellukset voidaan myöhemmin työtilasta viedä export komennolla JSON muodossa kopioituna tai tiedostoon tallennettuna. Samalla tavalla voidaan tuoda työtilaan uusi vuokaavio käyttämällä import komentoa. Tuodut ohjelmat saavat aina työtilaan uuden välilehden ja ne käynnistyvät vasta painettaessa deploy-nappia.

Node-RED tukee myös useamman työvuon käytön ja ohjelmat voivat keskustella keskenään, jos näin haluaa. Tämä mahdollistaa myös ajoympäristön normaalin toiminnan uusien ohjelmien tekemisen yhteydessä.

2.3 MQTT

Node-RED kehitys alkoi alun perin MQTT laitteiden välisestä kommunikaatiosta. MQTT-kommunikaatiota käytetäänkin monessa eri Node-RED sovelluksessa, koska Node-RED pystyy vastaanottamaan ja lähettämään MQTT-protokollalla dataa ilman ulkopuolisia työkaluja tai sovelluksia.

MQTT-protokolla kehitettiin vuonna 1999, kun IBM:llä työskennelleet Andy Stanford-Clark ja Arlen Nipper halusivat laajentaa MQ kommunikaatio teknologiaa. MQ on IBM:n viestintään keskittyvä kommunikaatio teknologia. MQTT-protokollan tarkoituksena oli kehittää yhteystapa, jolla voitiin ottaa yhteys öljyputkiin satelliittiyhteyden kautta. (HiveMQ Team 2015.)

MQTT teknologialle määritettiin useita eri vaatimuksia, jotka ovat edelleen kehittämisen periaatteina MQTT protokollalle:

1. Helppo käyttöönotto.
2. QoS (Quality of Service), tietoliikenteen luokittelu ja priorisointi.
3. Datakaistan käyttö mahdollisimman tehokkaaksi ja kevyeksi.
4. MQTT toimii kaikilla ohjelmointikielillä ja laitteilla.
5. Jatkuva yhteyden tarkkailu, jolloin yhteyden katkettua tiedetään mitä dataa ei käyttäjä ole vastaanottanut ja lähetetään ne käyttäjälle ensimmäisenä.

(HiveMQ Team 2015.)

MQTT julkaistiin IBM:n toimesta vuonna 2010 avoimena sovelluksena kaikkien käyttöön. Kolme vuotta myöhemmin MQTT otettiin OASIS Standardin alaisuuteen, jolloin protokolla laitettiin OASIS standardin mukaiseksi ja julkaistiin vuonna 2014. OASIS on avoin järjestö, jonka periaatteisiin kuuluvat eri standardien kehittäminen. Uusin standardisoitu versio MQTT-protokollasta on versio 5, joka mahdollistaa pilvessä olevien IoT-laitteiden kommunikaation ja kykenee myös käsittelemään kriittiseksi luokiteltua tietoa luotettavasti. (HiveMQ Team 2015.)

MQTT on viestintäprotokolla palvelimen ja asiakkaan välillä. Asiakas voi olla lähettävä tai vastaanottava laite. Protokolla on yksinkertainen, avoin ja helppo käyttöönottaa. Protokolla on siis erinomainen IOT-laitteissa, jolloin se käyttää vain vähän verkkokaistaa ja eikä kommunikaatio aiheuta koodinsa takia turhaa kuormitusta laiteympäristölleen. (OASIS Open 2015.)

MQTT-protokollassa itsenäiset laitteet lähettävät tietoa MQTT-palvelimelle ja palvelin lähettää MQTT-paketit niille, jotka ovat tilanneet kyseisen laitteen paketit. Tilaaminen määritellään MQTT-pyynnössä, johon palaan myöhemmin käytännön osuudessa. Tämä kommunikaatiotapa mahdollistaa turvallisemman kommunikaation laitteiden välillä, koska lähettäjä ei koskaan kommunikoi suoraan vastaanottajan kanssa. Samalla yksittäisen laitteen kuorma vähenee, koska se lähettää dataa vain yhdelle palvelimelle eikä usealle käyttäjälle.

MQTT-protokollaan käytetään laajasti myös Suomessa. Pääkaupunkiseudun julkisen liikenteen HSL (Helsingin seudun liikenne) käyttääkin MQTT-teknologiaa omien palveluiden toteuttamiseen. HSL jakaa myös MQTT dataa avoimesti muidenkin käyttöön, jolloin voi tehdä oman sovelluksen millä tahansa ohjelmointikielellä seuraamaan juna-, raitiovaunu tai bussiliikennettä. HSL MQTT-teknologiaan tutustutaan myöhemmin käytännön sovelluksena tässä opinnäytetyössä, jolloin seuraamme junaliikennettä.

2.4 JavaScript

Node.js pohjautuu JavaScriptin käyttöön saumattomasti backendin ja front-endin välillä. Node.js alun perin mahdollisti JavaScriptin käyttämisen myös sovelluksen back-end puolella. Node-RED pohjautuu Node.js ajoympäristöön, joten JavaScript on myös oleellista Node-RED:n käytössä. JavaScript ohjelmointia joutuu tekemään Node-RED:ssä, jos haluaa toteuttaa monipuolisemman sovelluksen.

David Flanagan (2020) mainitsee kirjassaan ”JavaScript the Definite Guide” JavaScriptin olevan kaikkein käytetyin ohjelmointikieli maailmassa. JavaScript on ollut käytössä vuodesta 1996 ja lähes kaikki internet-sivustot käyttävät JavaScriptiä. Melkein kaikki puhelimet, tietokoneet ja tabletit ovat varustettuja tulkitsemaan JavaScriptiä, jotta ne pystyvät näyttämään sovellukset ja verkkosivustot sellaisena kuin ne on tarkoitettu.

JavaScriptillä pystyy toteuttamaan interaktiivisia sivustoja, pelejä ja sovelluksia. Yksinkertainen JavaScript voi kysyä käyttäjän nimeä ja tämän jälkeen tervehtii käyttäjää syötteen perusteella. Yksikkömuunnokset, kuten valuuttakurssien laskeminen, onnistuu myös JavaScriptillä helposti. Tämän opinnäytetyön esimerkissä käytetään JavaScriptiä MQTT-

viestien karsimiseen ja muokkaukseen, koska välillä vain osa tiedosta on tarpeellista tai pakollista.

Tietoturvaongelmat ovat olleet JavaScriptin mainetta haittaava tekijä, mutta ohjelmointikielen kehittäminen ja internet-selaimien uudet turvatoiminnot estävät yleisimmät haitat mitä JavaScriptillä toteutetaan. Ennen tietoturvapäivityksiä sivuston koodilla pystyttiin muuttamaan selaimen toimintaa, jolloin kaikki tietoturvalliset välitiedostot menivät hakkereille. Tämä vaati käyttäjältä vain hyväksynnän pienen ponnahdusikkunan ilmoitukseen: ”Hyväksytkö Cross-Site Scriptingin tältä sivustolta”. Hyväksyntä mahdollisti hyökkääjälle pääsyn kaikkiin internet palveluihin mitä käyttäjä oli itse käyttänyt selaimella, kuten esimerkiksi pankkipalveluihin, sähköpostiin ja muihin arkaluontoisiin verkkosivuihin. (Veracode 2021.)

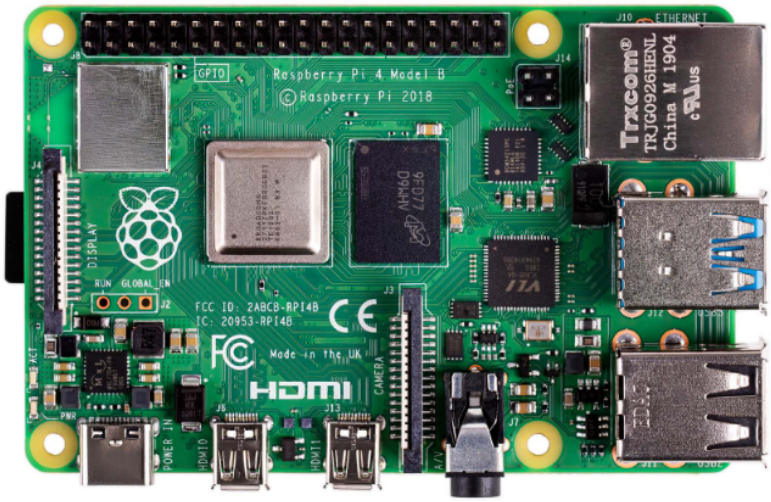
JavaScript on perusta Node-RED ohjelmointiin ja sen avulla voidaan muokata Node-RED moduulien toimintaa. Voit myös lisätä funktio-noden, jolloin ohjelmoidaan koko moduulin toiminta JavaScriptillä. Esimerkiksi MQTT ja JSON tietopaketit sisältävät usein paljon dataa, jolloin sen käsittely voi olla vaikeaa tai mahdotonta. Tiettyjen tietojen nappaaminen ja tallennus onnistuu JavaScriptin avulla. HSL MQTT-palvelu tuottaa valtavasti tietoa jatkuvasti, mutta pienen funktion avulla voit ottaa yksittäisen kulkuneuvon nopeuden tai suunnan. Voit jopa seurata keskinopeutta funktion ja tietokannan avulla.

En syvenny JavaScript-ohjelmointiin liikaa tässä opinnäytetyössä, koska aiheesta on runsaasti tietoa vapaasti saatavilla eri verkkosivujen, kurssien ja opetusvideoiden kautta.

2.5 Raspberry Pi

Raspberry Pi on edullinen yhden piirilevyn tietokone, jossa on tarpeeksi suorituskykyä pyörittämään eri Linux käyttöjärjestelmiä. Oletusarvoisesti laitetta käytetään Raspberry OS käyttöjärjestelmällä, joka avaa käyttäjälleen työpöydän ja joitakin oleellisia sovelluksia. Sovelluksia voi hakea lisää Raspberryn oman asennussovelluksen kautta tai suoraan komentorivikomennoilla.

Raspberryä on perinteisesti käytetty opettamaan lapsille ohjelmointia ja tietokoneen käyttämistä. Aikuisille harrastelijoille on myös omat projektinsa, kuten http-palvelin tai sääaseman tekeminen käyttäen ulkoisia sensoreita.



Kuva 4. Raspberry Pi 4 model B. (Raspberry Pi Trading Ltd., 2019)

Raspberry otetaan käyttöön laittamalla haluttu käyttöjärjestelmä muistikortille tai USB-medialle. Tietokoneen käynnistyessä se lukee muistikortilta tai USB-mediasta käyttöjärjestelmän tiedot ja käynnistää sille asennetun käyttöjärjestelmän. Laite menee suoraan työpöydälle ja pyytää vain muutamia lisätietoja käyttöjärjestelmälle, mutta muuta asentamista ei vaadita. Raspberry on muuttunut tehokkaammaksi jatkuvasti ja pystyykin jo pyörittämään vaativampia ohjelmia melko helposti.

Uusin Raspberry Pi 4 model B julkaistiin 2019, joka noudattaa Raspberryn tavanmukaista yhden piirin tietokoneen mallia. Raspberry Pi 4 on edullinen, mutta käyttöä varten pitää ostaa vielä mm. virtalähde ja micro-SD muistikortti. Raspberry ja muut jälleenmyyjät myyvät myös pakettia, johon kuuluvat kaikki tarpeelliset osat käyttöä varten. Myynnissä on myös näppäimistö, jonka sisään on rakennettu valmiiksi Raspberry tietokone.

Raspberry soveltuu Node-RED alustaksi erinomaisesti sen pienen koon ja matalan virtakulutuksen ansiosta. Raspberry vaatii vain virtakaapeliksi USB-C kaapelin ja 5.1 V 1.2–3.0

A virtalähteen toimiakseen vakaasti. USB-C liitäntä on yleinen jo monessa puhelimessa ja tietokoneessa, joten voit käyttää myös vaihtoehtoista virtalähdettä. Raspberry osaa ilmoittaa, jos virtalähteessä on ongelmia tai se on riittämätön. Virtalähteen vaatimukset kasvavat, jos laitteen prosessorin kellotaajuksia säädetään suuremmaksi eli ylikellotetaan. Prosessorin tehon muokkaaminen on hieman monimutkaisempaa ja vaatii erillisen jäähdytys ratkaisun.

Oletuksena on, ettei Raspberryssä ole erillistä jäähdytystä, koska se pyörii noin 50–55 asteen lämpötilassa ilman rasitusta pienessä kotelossa. Jos lämpötila on yli 60 astetta jatkuvasti, voi tällöin harkita jäähdytyksen hankkimista, koska pitkittynyt kovempi lämpötila voi lyhentää prosessorin elinikään.

2.5.1 Raspberry Pi 4 model B ominaisuudet

- 64 bittinen Cortex-A72 (ARM v8) prosessori
- 2GB, 4GB or 8GB LPDDR4-3200 SDRAM
- WIFI 2.4 GHz and 5.0 GHz IEEE 802.11ac
- Bluetooth 5.0, BLE
- RJ45-verkkoportti
- Kaksi USB 3.0 porttia
- Kaksi USB 2.0 porttia
- Kaksi MIPI DSI display porttia (micro-HDMI)
- Micro-SD kortinlukija
- Stereo kuulokeliitäntä
- USB-C virtaliitin

(Raspberry Pi Foundation, s.a)

Liitännät mahdollistavat Raspberryn käyttämisen melkein missä ympäristössä tahansa, kuten työasemana, palvelimena, IOT-laitteena tai vaikka palomuurina. Raspberry Foundation ylläpitää sivustoa projekteille, josta voi löytää itselleen sopivan harjoitteluprojektin tai vaikka lapselle sopivan opiskeluprojektin.

2.5.2 Raspberryn käyttöjärjestelmät

Raspberry Pi tietokoneella toimii Linux-pohjainen Raspberry (ent. Rasbian) käyttöjärjestelmä. Tällä hetkellä käyttöjärjestelmä on 32 bittinen, mutta Raspberry pystyy pyörittämään myös 64-bittisiä käyttöjärjestelmiä. Ubuntu julkaisi vuonna 2020 oman työpöytä käyttöjärjestelmänsä 64-bittisenä Raspberrylle. Ubuntun käyttöjärjestelmä on erittäin mukava käyttää, koska sen graafinen käyttöliittymä on miellyttävä silmälle. Ongelmana on

hieman keskeneräinen ympäristö, joka ei ole vielä optimoitu Raspberrylle, joten käytön yhteydessä voi olla hidastelua. Toivottavasti päivitykset saavat Ubuntun myös vaihtoehtoiseksi käyttöjärjestelmäksi. Tarjolla on myös lukuisia muita käyttöjärjestelmiä, joiden toimintaan emme nyt tutustu.

Raspberry Pi:n prosessoriteknologia on vaikeuttanut käyttöjärjestelmien kehittämistä laitteelle, koska ARM prosessorit toimivat eri tavalla kuin yleisimmät AMD ja Intel prosessorit. ARM prosessorit ovat yleistyneet viime vuosina puhelimien ja tablettien käytössä. Uusimmat Applen tietokoneet pohjautuvat ARM prosessoriteknologiaan.

Aloittelijalle suositellaan Raspberryn omaa käyttöjärjestelmää, koska sitä päivitetään jatkuvasti ja se toimii myös parhaiten Raspberryllä.

2.5.3 Etäyhteydet Raspberryllä

Sovellukset, jotka ovat toteutettu Node-RED:llä ovat yleensä jatkuvassa käytössä, tällöin huomaamaton ja hiljainen tietokone on hyvä ratkaisu kotiolosuhteissa. Sovellusten tekoa varten olevaa käyttöliittymää voi käyttää lähiverkossa miltä tahansa koneelta tai jopa tablettitietokoneelta, koska käyttöliittymä on toteutettu JavaScriptillä ja pyörii selainympäristössä.

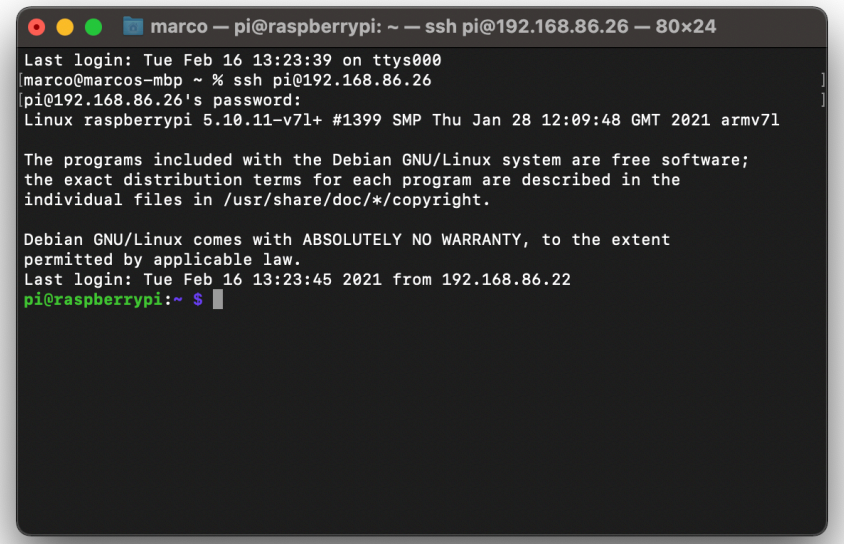
SSH-etäyhteyden kautta voidaan asentaa Node-RED sovellus ja sen vaatimat riippuvuudet eli ohjelmat mitä Node-RED tarvitsee toimiakseen. Asennuksen jälkeen suosittelen asettamaan Node-RED palvelun käynnistymään automaattisesti Raspberryn käynnistyksen yhteydessä, jolloin sovellukset myös pysyvät käynnissä. Tällöin sähkökatkokset tai Raspberry laitteen siirtäminen ei vaikuta ohjelmiston toimintaan pitkäaikaisesti. Kaikki nämä voidaan toteuttaa etäyhteydellä.

Node-RED asennuksen jälkeen voit navigoida tietokoneen IP-osoitteeseen, jolloin ohjelmointi työtila avautuu käytettäväksi. Node-RED sallii myös yhteyksiä lähiverkon ulkopuolelle, mutta tällöin pitää olla varma käyttöliittymän turvallisuudesta. Oletusarvoisesti kaikki lähiverkon tietokoneet voivat avata työtilan ja tehdä muutoksia ohjelmiin, jolloin se on melko turvaton oletusarvoillaan.

Turvallisuus periaatteet käydään läpi tämän opinnäytetyön käytännön osuudessa, jossa selostan kuinka eri turvametodit pyrkivät pitämään laitteesi turvallisena.

2.5.4 SSH-yhteys

Raspberry käyttöjärjestelmä on käytettävissä myös etäyhteyksillä, jolloin laite ei vaadi muuta kuin virtalähteen. Raspberryyn saa Secure Shell eli SSH yhteyden lähiverkon kautta helposti. Kuvassa (kuva 5) on muodostettu SSH yhteys Applen OSX käyttöjärjestelmän terminaalista, SSH käyttö OSX ympäristössä ei vaadi erillisiä asennusta tai ohjelmaa.



Kuva 5. SSH-yhteyden muodostaminen Raspberryyn.

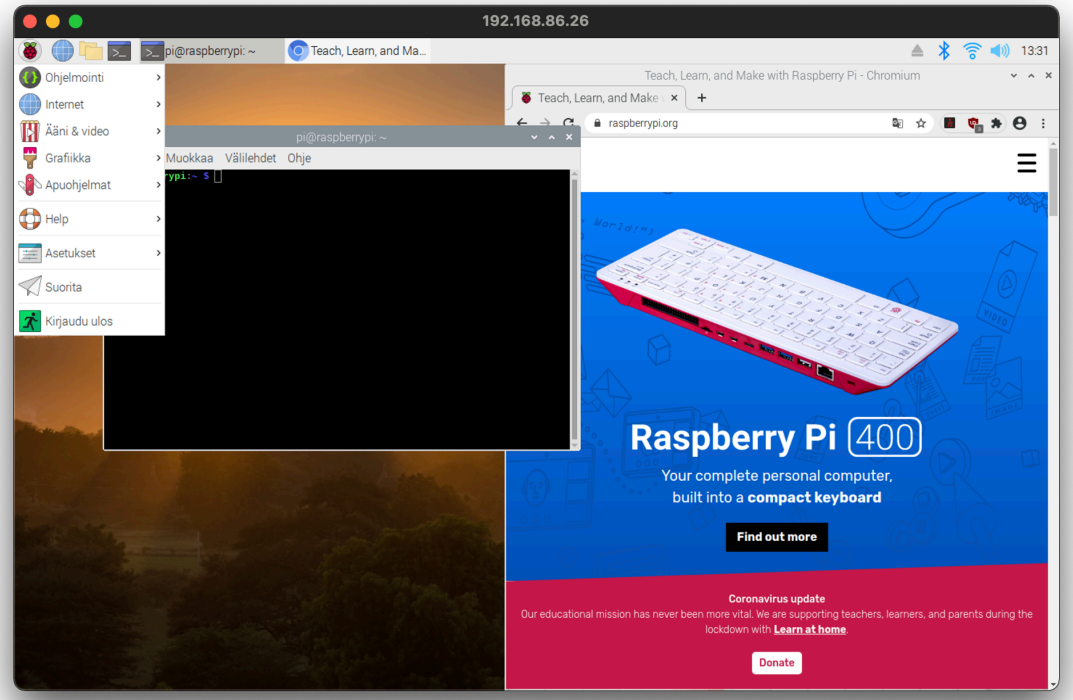
SSH-protokolla on suosittu ja tehokas ohjelmistopohjainen lähestymistapa verkkoturvallisuuteen. SSH yhteys salaa lähetettävän datapaketin ja vastaanottaja purkaa salauksen datapaketista paketin saavuttua. Salaus on toteutettu tehokkailla algoritmeillä ja on tarpeeksi turvallinen, että sitä voidaan käyttää myös kriittisissä järjestelmissä isoissa yrityksissä. (Barrett, Silverman & Byrnes, 2005.)

Pelkän SSH yhteyden kautta pystyy tekemään lähes kaiken, kuten ohjelmien asentaminen, laitteen päivittäminen ja jopa uudelleen käynnistykset. SSH yhteys mahdollistaa tekstipohjaisen komentoriviyhteyden, jolloin tietokonetta voidaan hallita Linux komentojen avulla, mutta SSH pitää kytkeä päälle Raspberryn asetuksista. Raspberryn käyttöönoton yhteydessä on hyvä tarkistaa SSH toimivuus. Muista tarkistaa laitteen IP-osoite, jotta yhteyden pystyy muodostamaan.

Esimerkiksi SSH yhteyden kautta asennan Node-RED:in, mutta palaamme siihen myöhemmin. Asennuksen jälkeen voidaan laittaa sovelluksen käynnistymään laitteen uudelleenkäynnistyksen yhteydessä, jolloin Node-RED pysyy käytössä aina laitteen ollessa päällä. Työtilat voidaan kytkeä pois päältä yksitellen, jos näin haluaa tehdä.

2.5.5 Etätyöpöytä

Etätyöpöytäyhteys toimii myös Windows, Mac ja Linux tietokoneista, jos asentaa Raspberryyn esimerkiksi avoimenlähdekoodin Xrdp sovelluksen. Etätyöpöytä yhteyden käyttäminen on myös toimiva tapa hallita Raspberryä, mutta se kuormittaa hieman enemmän verkkoa ja vaatii Raspberryä avaamaan työpöytäversion käytettäväksi. Ilman näyttöä laite toimii täysin CLI (Command Line Interface) tilassa eli pelkässä komentorivi tilassa.



Kuva 6. Etätyöpöytäyhteys Raspberryyn.

Kuvassa (kuva 6) olen muodostanut etätyöpöytäyhteyden Raspberryyn Microsoft Remote Desktop ohjelmalla Mac OSX käyttöjärjestelmästä. Etätyöpöytä-työkaluja löytyy toki enemmän ja parempiakin, mutta Xrdp on osoittautunut yksinkertaiseksi ja helpoksi asentaa. Sen toimivuuden olen myös testannut monelta eri alustalta, kuten Windows, Mac ja Android. Jokaiselle alustalle löytyy oma sovellus etätyöpöydän käyttöön.

Etätyöpöytä yhteys on voi olla parempi niille, joille Linux käyttöjärjestelmä ei ole kovinkaan tuttu. Työpöydältä voi säätää helposti Raspberryn asetuksia tai asentaa ohjelmia ilman CLI:n käyttöä. Toiminta on hyvin samankaltainen kuin Mac OSX ja Windows käyttöjärjestelmissä, mutta valitettavasti suorituskyky on laitteen hinnan mukainen työpöytäkäytössä. Ohjelmointiin, opetukseen ja palvelinkäyttöön laite riittää mainiosti.

3 Node-RED-sovelluksen käyttöönotto Raspberry Pi tietokoneessa

3.1 Johdanto

Tässä osiossa tutustumme ensin Raspberryn käyttöönottamiseen, järjestelmän päivittämiseen, ohjelmien asentamiseen, tietoturvallisuuteen ja etäkäyttöön. Teemme myös pienimuotoinen tutustuminen Linux käyttöympäristöön, jotta voimme käyttää Raspberryä tehokkaasti. Linuxin työpöytäsovelluksiin tai muihin ominaisuuksiin emme tässä ohjeistuksessa syvenny. Käyttöjärjestelmänä Linux on loistava, mutta siihen kannattaa tutustua itsenäisesti kirjallisuuden, kurssien tai internetissä olevien videoiden avulla.

Ideana tälle opinnäytetyölle oli tuottaa materiaalia, jossa käydään läpi, kuinka otamme käyttöön Raspberry Pi tietokoneen ja Node-RED-sovelluksen. Haluasin kuitenkin ensin käydä läpi periaatteita mitä projektissa toteutetaan ja miksi valitsin kyseiset lähestymistavat tähän opinnäytetyöhön. Toivoisin lukijan saavan hieman paremman tietotaustan, kun siirrymme teoriasta toteutukseen, vaikka uskonkin kiinnostuksen olevan lähinnä vain asiaan perehtyneillä. Tehdessäni käytännön osiota tässä opinnäytetyöstä, huomasin myös itse oppineeni uusia toimintatapoja projektin toteutukseen. Tietoturva on ollut mielenkiintoni kohde siitä lähtien, kun tutustuin ensimmäisen kerran Raspberryyn ja Node-RED:iin 2020 syksyllä. Pystyin syventymään siihen hieman enemmän tämän projektin myötä, mutta pääpaino on kuitenkin ohjelmoinnissa ja miten se onnistuu pienellä IOT-laitteella.

Käytännön osuus tässä opinnäytetyössä etenee kronologisesti, eli kappaleet ovat järjestetty sen mukaan, kuinka toteutin ne työtä tehdessäni. Projektia varten tein uuden Raspberry OS muistikortin eli aloitin puhtaalta pöydältä, jolloin pystyin etenemään kappaleissa samaan tahtiin lukijan kanssa. Kappaleet eivät välttämättä edellytä edellisen kappaleen sisällön toteuttamista päästäkseen eteenpäin. Päivitykset ovat suotavia, mutta eivät pakollisia. Tietoturvaan liittyvät eivät ole lainkaan pakollisia, mutta ovat mielestäni osa tämän opinnäytetyön ideaa. Jos aiot toteuttaa laitteen, joka on jatkuvassa internet yhteydessä, silloin suosittelen turvaamaan laitteen.

Lähdin projektia toteuttamaan tyhjästä, koska halusin myös havaita muutokset ohjelmiin tai komentoihin tai käytänteisiin. Esimerkiksi Node-RED asentaminen on helpottunut huomattavasti aiemmasta. Halusin myös varmistua ohjeiden ajantasaisuudesta, ohjeissa mainitsen välillä tarkistamaan uusimman komennon tai ohjelman tekijän verkkosivuilta. Ohjelmistot ja Raspberry kehittyy jatkuvasti ja voi olla, että viikon kuluttua tämän opinnäytetyön valmistumisesta julkaistaankin jo 64-bittinen versio Raspberry OS käyttöjärjestelmästä, jolloin ohjeet pitäisi jälleen tarkistaa.

3.2 Raspberryn käyttöönotto

Raspberryn Pi tietokonetta myydään useassa eri kokoonpanossa, jolloin itse Raspberry Pi on sama, mutta mukana tulevat osat eroavat toisistaan. Joissain tapauksissa paketin mukana tulee myös kotelo, mutta Raspberryn käyttäminen ei edellytä edes kotelon käyttämistä. Perusasennusta varten tarvitset micro-HDMI kaapelin, näytön, näppäimistön, hiiren ja SD-muistikortin. Muistikortin kirjoittamista varten tarvitset SD-muistikortin lukijan.

Jos paketissa ei tule valmiiksi asennettua käyttöjärjestelmää, joudut asentamaan sen itse muistikortille. Onneksi Raspberry OS käyttöjärjestelmän asentaminen on helppoa. Raspberry Pi verkkosivustolta löytyy Raspberry Pi Imager -sovellus, jolla pystyy kirjottamaan SD muistikortille käyttöjärjestelmän kokonaisuudessaan. Sovellus pystyy noutamaan internetistä myös muita käyttöjärjestelmiä ja kirjoittamaan ne muistikortille, jolloin muidenkin käyttöjärjestelmien asentaminen on helppoa. Itselläni on useampi muistikortti ja pystyn vaihtamaan tarvittaessa käyttöjärjestelmää vaihtamalla vain muistikortin.

Myöhemmin voit myös asentaa Raspberryn perinteiselle kiintolevylle tai uudemmalle SSD-levylle, mutta tämä vaatii jo melko paljon perehtymistä Raspberryn toimintoihin.

Ensimmäisen käynnistyksen yhteydessä valitaan maa-asetukset eli aikavyöhyke ja käyttöjärjestelmän kielivalinnat. Samalla asetetaan käyttöjärjestelmän salasana, jolla kirjaudutaan koneelle paikallisesti tai verkon kautta. Node-RED ei vaadi kyseistä salasanaa, koska vuoeditoriin pääsee oletusarvoisesti ilman salasanaa ja toimii Raspberryn Node.js palvelimen kautta. Raspberry hakee käyttöjärjestelmälle uusimmat päivitykset ja asentaa ne ensimmäisen, kun maa-asetukset ja salasana on määritelty.

Ensimmäinen vaihe koostuu siis käyttöjärjestelmän kirjoittamisesta muistikortille ja käynnistyksen yhteydessä tehtävät määritykset ja päivitykset. Raspberry OS opastaa näiden tekemiseen, jolloin erillistä ohjetta niiden toteuttamiselle ei tarvita. Ainoastaan SSH käynnistämine vaatii erikseen hyväksynnän Raspberry Pi Configuration -sovelluksesta. Asetukset – Raspberry Pi Configuration – Interfaces, välilehdestä löytyy SSH rivi, johon laitetaan enable päälle.

3.3 Raspberryn ohjelmien asentaminen ja LXTerminal

Raspberryn OS on Linux pohjainen käyttöjärjestelmä eikä siinä ole automaatista päivitystä oletuksena eikä se muistuta niistä erikseen. Päivittäminen tehdään ajoittain manuaalisesti tai automatisoidaan esimerkiksi crontab-sovelluksen kautta. Crontabiin kannattaa tutustua, jos aikoo käyttää Linux tietokoneita enemmän myöhemmin.

Raspberry OS ei sisällä päivitystyökalua, esimerkiksi Ubuntulla löytyy oma päivitysohjelma, mutta Raspberry OS on toiminnaltaan hieman erilainen kuin Ubuntu. Tavoitteena on tietokone, jonka voi nostaa hyllyn päälle ja unohtaa, eikä päivitykset muuta sen toimintaa. Automaattiset päivitykset voivat vaatia koneen uudelleen käynnistyksen ja saattaa täten sammuttaa palveluita, joita tarvitsee laitteella jatkuvasti. Meidän tapauksessamme päivitykset eivät ole ongelma ja voimme päivittää laitteen.

Raspberry OS LXTerminal mahdollistaa Linux komentojen ajamisen tietokoneella ja tätä kautta suurin osa toiminnasta tapahtuu. LXTerminal ja SSH yhteydellä avattava terminaali toimivat samankaltaisesti. SSH yhteydestä puuttuu joitakin ominaisuuksia mitä sisältyy LXTerminaliin, mutta päivittäessä ja ohjelmia asentaessa eroavaisuuksia ei huomaa.

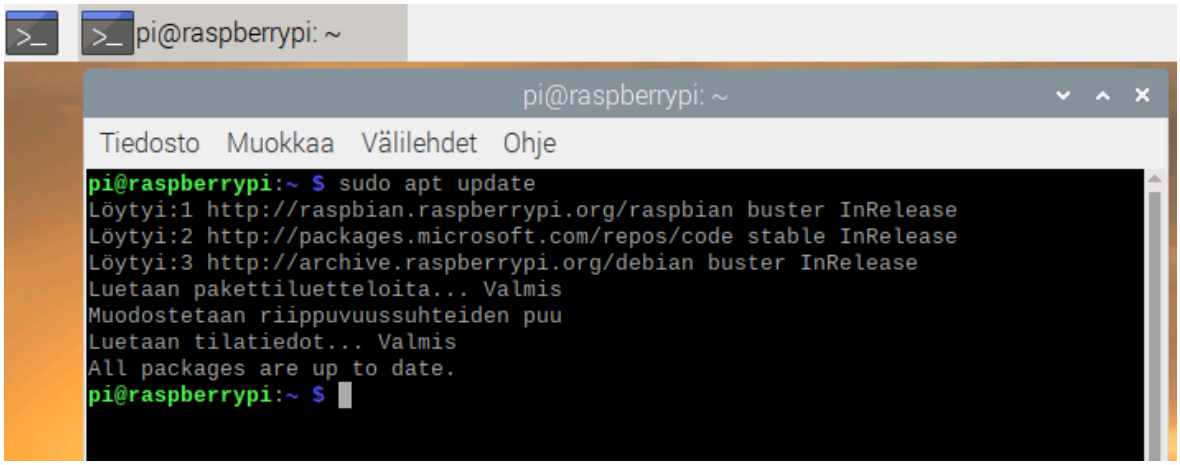
Päivitykset asentuvat melkein automaattisesti eivätkä vaadi käyttäjältä kuin oikean komennon ja hyväksynnän asennukselle. Ohjelmistojen asentaminen on Linux ympäristössä hieman erilainen, koska voit asentaa ohjelmat asennustyökalulla työpöydän kautta tai terminaalista oikealla komennolla. Helpoiten saat oikean latausohjeen sovelluksen kehittäjän verkkosivustolta, jolloin voit kirjoittaa tai kopioida komennon suoraan terminaaliin. Voimme esimerkiksi asentaa meille Xrdp etätyöpöydän SSH:n tai terminaalin kautta suoraan. Ensimmäisenä päivitämme pakettiluettelon ”***sudo apt update***”-komennolla, jonka jälkeen jatkamme komennolla ”***sudo apt-get install xrdp”***. Terminaali hakee sovelluksen tiedot verkosta ja varmistaa vielä haluatko asentaa sovelluksen, valitse ”K” eli ”kyllä”.

Oikean sovelluksen löytäminen voi olla vaikeaa Raspberryn omalla ”Add/Remove Software”-sovelluksella. Esimerkiksi sillä löytyy tällä hetkellä neljä eri sovellusta hakusanalla: ”node-red”, joista vain yksi on käyttämämme Node-RED. Parilla komennolla saamme saman tehtyä paljon helpommalla.

Tässä opinnäytetyössä on käytössä konsolikomennot jatkuvasti, joten viittaan komennoilla aina terminaaliin syötettävinä komentoina. Ilmoitan erikseen komennot, jotka tulee syöttää työpöydän tai etätyöpöydän kautta, tällöin tarvitsemme työpöytäsovelluksen tarjoamia laajennuksia. SSH-yhteydellä voimme melkein toteuttaa kaiken.

3.3.1 Raspberry OS päivittäminen

Kuvassa (kuva 7) nähdään miltä terminaali näyttää käytännössä. Terminaalissa on ajettuna komento, joka tarkistaa päivitysten saatavuuden. Linux käyttöjärjestelmän pakettiluettelo eli virallinen lista ohjelmista tulee noutaa komennolla: ”***sudo apt update***”.



Kuva 7. Raspberry OS LXTerminal käytössä.

Toiminto on hyvin samankaltainen kuin Windowsin päivitysten tarkistamisessa, tosin Windows päivitykset yleensä tarkistetaan päivityssovelluksen kautta. Komento tarkistaa vain onko saatavilla uusia päivityksiä laitteella oleviin sovelluksiin, ajureihin tai käyttöjärjestelmään. Kun komento on ajettu ja se on noutanut listan uusista päivityksistä, voidaan tämän jälkeen ajaa päivitykset.

Päivittämiseen on monia vaihtoehtoja, mutta suosittelen Raspberryn omia ohjeita (Raspberry Pi Foundation 2021b), jossa mainitaan terminaalikomento: ”***sudo apt full-upgrade***”. Tämä päivittää kaikki laitteen asennetut sovellukset uusimpiin versioihin. Komento päivittää myös ohjelmien sidonnaisuudet eli sovellukset, jotka käyttävät toisen sovelluksen ominaisuuksia toimiakseen. Päivitysohjelma kysyy vahvistuksen päivitysten asennusta varten, kertoo mitä on päivittämässä ja kuinka iso tiedosto pitää ladata. Vahvista päivitysten asentaminen kirjoittamalla ”Y” tai ”K”, riippuen käyttöjärjestelmän kielestä.

Kyseinen ”full-upgrade” ei kuitenkaan päivitä käyttöjärjestelmää uudempaan versioon, kuitenkin ohjelmat ja tietoturva pysyvät ajan tasalla. Komennolla: ”***sudo apt dist-upgrade***”, voidaan päivittää käyttöjärjestelmä uusimpaan, mutta tällöin ohjelmien toiminta voi muuttua. Päivittäminen voi viedä hetken, mutta päivitykset parantavat tietokoneen toimintaa ja ovat olennainen osa tietoturvallisuutta.

3.4 Raspberryn turvallisuus

Raspberry OS on tietoturvallinen, jos siihen on uusimmat päivitykset asennettuna ja tässä kappaleessa tehdyt perusasetukset ovat kunnossa. Aina voit tietenkin parantaa turvallisuutta lisäämällä esimerkiksi salausavaimen SSH-kirjautumiseen. Tietoturvallisuus kannattaa suhteuttaa siihen ympäristöön, jossa tietokone sijaitsee. Yrityksen tai valtion tiloissa tietenkin Raspberry omaan suljettuun verkkoon, SSH-avaimet kirjautumiseen ja valitut päivitykset vain asennetaan, mutta tässä projektissa meille riittää normaalit turvallisuus periaatteet laitteen käytössä. Muista kuitenkin laitteen olevan mainio ponnahduslauta verkon muihin tietokoneisiin, jos tietoturva ei ole toiminnassa.

Raspberry OS toimii niin kauan kuin SD-kortin sisältö pysyy ehyenä. Tiedon eheys on aina vaakalaudalla, kun käytetään muistikortilta latautuvaa käyttöjärjestelmää. Jos aikoo käyttää Raspberryä pidempää tietenkin on hyvä varmistaa sen käyttökelpoisuus pidemmälle ajalle. Muistikortin voi kopioida ja vaihtaa tarvittaessa, jolloin laite palautuu edellisen kopioinnin tilaan, mutta SSD levyt ovat luotettavampia kuin SD-muistikortit. SD-muistikortti on altis virtapiikeille, ja tiedostojärjestelmä saattaa vaurioitua jopa virrankatkaisusta. Suosittelen itse käyttämään terminaalin komentoa ”***sudo shutdown***”, jolloin laite sammuu minuutin päästä. Uudelleen käynnistys toimii komennolla ”***sudo reboot***”. Kyseiset komennot sulkevat käyttöjärjestelmän hallitusti.

Tiedon saatavuus on Raspberryn parhaita ominaisuuksia, koska voit jättää laitteen päälle ja se on aina käytettävissä. Itse olen käyttänyt Raspberryä tiedostopalvelimena, johon voin langattoman lähiverkon kautta siirtää kuvat ja videot puhelimesta myöhempää katselua varten. Langattoman lähiverkon kautta tiedonsiirto voi olla melko hidasta, mutta nopeus ei välttämättä ole tärkein ominaisuus vaan kuvien varmuuskopiointi.

Raspberryn käyttöjärjestelmän turvaamme luomalla uuden käyttäjätunnuksen, estämme komentojen ajamisen ilman salasanaa ja torjumme ulkopuoliset yhteydet tietokoneelle palomuurin avulla. Pienillä toimilla voimme estää hakkerin pääsyn laitteen hallintaan, vaikka hakkeri pääsisi syöttämään viallisen verkkosivuston tai tietokannan kautta komentoja koneen terminaaliin. Terminaalin avulla hakkeri voi halutessaan poistaa käyttäjätunnuksia, luoda uusia, kopioida laitteen sisällön tai jopa muuttaa laitteen toimintaa täysin erilaiseksi kuin aiemmin. Hakkerit ovat aiemmin ottaneet käyttöön IOT-laitteita toteuttaakseen Botnet-hyökkäyksiä, jolloin tietokone valjastetaan hakkereiden omia hyökkäyksiä varten.

3.4.1 Uusi tunnus

Turvallisuuden takaamisen voimme aloittaa luomalla oman käyttäjätunnuksemme. Raspberry Pi tietokoneissa tulee automaattisesti käyttäjätunnukseksi ”pi”, jolloin hakkerin tarvitsee vain arvata oikea salasana ja pääsee tietokoneen juureen käsiksi.

Aloitetaan luomalla uusi tunnus konsolikomennolla: ”***sudo adduser kayttaja***”. Komennossa käyttäjä on ilman erikoismerkkejä, voit tietenkin luoda käyttäjän omalla nimellä tai nimimerkillä. Seuraavaksi annamme käyttäjälle oikeudet tehdä asioita tietokoneella.

Käyttäjälle annettaan järjestelmänvalvojan oikeudet ja kaikki muut tarpeelliset oikeudet. Terminaaliin syötetään seuraava pitkä komento kokonaisuudessaan: ”***sudo usermod -a -G adm,dialout,cdrom,sudo,audio,video,plugdev,games,users,input,netdev,gpio,i2c,spi kayttaja***”. Komento antaa melko paljon oikeuksia käyttäjälle, koska tarkoituksemme on korvata ”pi”-tunnus uudella ”kayttaja”-tunnuksella. Raspberryn omilla verkkosivuilla on kyseinen komento saatavilla, jos haluat sen kopioida suoraan. (Raspberry Pi Foundation 2021c.)

Seuraavaksi testaamme uuden käyttäjätunnuksen toiminnan komennolla: ”***sudo su – kayttaja***”. Komento pitäisi onnistua, jos käyttäjä kuuluu ”sudo”-ryhmään. Aiemmassa komennossa nostimme käyttäjän mm. ”sudo”-ryhmään, jolloin käyttäjä pystyy tekemään ”sudo”-komentoja. Voimme nyt sammuttaa ”pi”-käyttäjän prosessit taustalta: ”***sudo pkill -u pi”*** ja poistaa koko tunnuksen komennolla ”***sudo deluser pi***”. Joskus ”pi”-tunnusta ei voi tai ei kannata poistaa niin silloin kannattaa vain vaihtaa salasana ja lopettaa prosessit, jonka jälkeen siirtyy käyttämään uutta käyttäjätunnusta. Jos olet asentanut jo jotain ”pi”-tunnukselle, se lakkaa toimimasta, kun tunnuksen prosessit sammutetaan. (Raspberry Pi Foundation 2021c.)

Seuraavaksi laitamme ”sudo”-komennon vaatimaan käyttäjältä salasanan, jolloin hakkeri ei pääse muuttamaan laitteen asetuksia, vaikka pääsisi syöttämään komentoja konsoliin. Komennolla: ”***sudo visudo /etc/sudoers.d/010_pi-nopasswd***”, avaamme nano tekstieditorilla muokattavaksi käyttäjätunnusten oikeuksia ajaa komentoja. Lisätään seuraava riviohjelmaan: ”***kayttaja ALL=(ALL) PASSWD: ALL***”, joten kaikki komennot vaativat salasanan. Jos et ole poistanut ”pi”-käyttäjää, lisää myös sille samat muutokset tiedostoon. Tallenna muutokset painamalla crtl+o ja poistu editorista painamalla crtl+x. Nyt Raspberry vaatii salasanan aina ”sudoa” käytettäessä, jolloin se on paljon turvallisempi. (Raspberry Pi Foundation 2021c.)

3.4.2 Palomuuri

Raspberryssä on valmis palomuuri nimeltään iptables/netfilter, mutta sen käyttäminen voi olla monimutkaista varsinkin uudelle käyttäjälle. Asennamme Raspberryyn yksinkertaisemman, mutta tehokkaan UFW (Uncomplicated Firewall) palomuurin, jolloin voimme hallita mistä tietokoneeseen pääsee käsiksi ja mitä sieltä kuuluu saada ulos.

Aloitamme komennolla: ”***sudo apt update***”, jos pakettien päivityksestä on aikaa. Jonka jälkeen voimme asentaa palomuurin: ”***sudo apt install ufw***”. Palomuuri sovelluksen asennuksen jälkeen voimme määrittää sääntöjä palomuurille, mutta teemme vain muutaman tätä opinnäytetyötä varten.

Ensin sallimme SSH liikenteen komennolla: ”***sudo ufw allow ssh***”, sudo antaa oikeudet muokata palomuurin eli UFW sääntöjä, allow sallii tietyn yhteyden palomuurin läpi ja ssh kertoo palomuurille mikä palvelu tarvitsee pääsyn koneelle. SSH komennossa kertoo UFW-sovellukselle, että haluan käyttää SSH-yhteyttä ja UFW tietää avata portin 22. Voit myös kirjoittaa: ”***sudo ufw allow 22***”, joka toteuttaa saman asian. Huomaa ettei palomuuri ole vielä päällä, joten emme katkaise vahingossa yhteyttämme tietokoneelle. (Raspberry Pi Foundation 2021c.)

Seuraavaksi sallimme käytön lähiverkossa etätyöpöydälle ja Node-RED vuoeditoriin. Jos myöhemmin haluat avata Node-RED sovellusta ulkoverkkoon, muista muokata UFW asetukset sallimaan yhteyden. Seuraavaksi syötämme komennon: ”***sudo uwf allow from 192.168.86.0/24 to any port 3389***”. Tällä komennolla annamme oikeuden Xdrp etätyöpöytäsovellukselle päästä palomuurin läpi, mutta vain lähiverkossa toimiville laitteille. Tämän voit myös toteuttaa komennolla: ”***sudo ufw allow 3389***”, jolloin portti on auki kaikkialle. Teemme vielä saman komennon Node-RED vuoeditorille: ”***sudo uwf allow from 192.168.86.0/24 to any port 1180***”, jolloin vuoeditoriin ei pääse kuin lähiverkosta. (Raspberry Pi Foundation 2021c.)

Voimme myös lisätä komennon: ”***sudo ufw limit ssh/tcp***”, jolloin ”brute-force” hyökkäykset koneelle estetään. ”Brute-force”-komento nimensä mukaisesti yrittää arvailla salasanaa loputtomasti, kunnes osuu oikeaan. ”Limit” rajaa yritysten määrän kuuteen kirjautumisyritykseen per 30 sekuntia. (Raspberry Pi Foundation 2021c.)

Palomuuri-sovelluksella olemme nyt sallineet vain haluamamme tiedon kulkevat Raspberryltä ulos. Palomuurin tilan voi tarkistaa komennolla: ”***sudo ufw status***”. Status komento

tulostaa palomuurin tilan ”inactive/active” ja porttikohtaiset ohjeet palomuurille. Kun asetukset näyttävät olevan kunnossa, voimme käynnistää palomuurin komennolla: ”***sudo ufw enable***”. SSH yhteys, Node-RED vuoeditori ja Xdrp-etätyöpöytä ovat nyt turvattuja.

3.5 Node-RED asentaminen

Aiemmassa kappaleessa ajettu komento ”sudo apt update” päivitti samalla koko ohjelmaluettelon eli haki myös valmiiksi meille Node-RED uusimman tarjolla olevan version ladattavaksi. Mutta Node-RED:in verkkosivusto tarjoaa myös komennon, jonka avulla saa Raspberrylle asennettua Node.js ja Node-RED yhdellä komennolla. Komento myös hakee Raspberry Pi koneelle omia moduuleja valmiiksi, kuten GPIO pinnien käyttämistä varten omat nodet vuoeditoriin.

```
Running Node-RED install for user pi at /home/pi on raspbian


This can take 20-30 minutes on the slower Pi versions - please wait.

  Stop Node-RED                       ✔
  Remove old version of Node-RED      ✔
  Remove old version of Node.js       ✔
  Install Node.js LTS                 ✔   Node v12.20.2   Npm 6.14.11
  Clean npm cache                     ✔
  Install Node-RED core               ✔   1.2.9
  Move global nodes to local          -
  Install extra Pi nodes              ✔
  Npm rebuild existing nodes          ✔
  Add shortcut commands               ✔
  Update systemd script               ✔


Any errors will be logged to   /var/log/nodered-install.log
All done.
  You can now start Node-RED with the command  node-red-start
  or using the icon under   Menu / Programming / Node-RED
  Then point your browser to localhost:1880 or http://{your_pi_ip-address}:1880
```

Kuva 8. Node-RED asennuspaketin sisältö.

Komento sovellusten asentamista varten löytyy sivustolta Node-RED:in Raspberry Pi sivustolta. En viittaa suoraan komentoon, koska sen voi helposti tarkistaa kyseiseltä sivustolta. Yleensä Linux asennuskomennot kannattaa kopioida suoraan terminaaliin, jotta ei vahingossa lataa koneellensa vahingollisia sovelluksia.

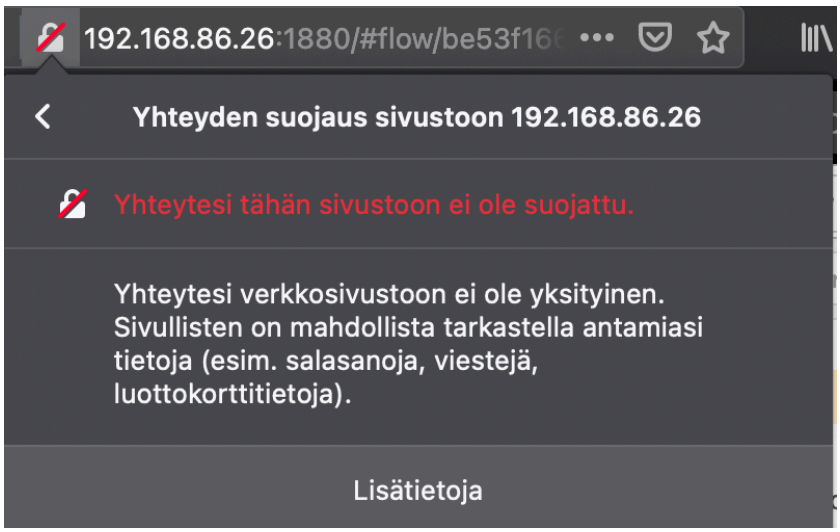
Kuvassa (kuva 8) voidaan havaita asennusohjelman asentaneen meille nyt valmiiksi uusimman Node.js, Npm ja Node-RED versiot, joten voimme aloittaa Node-RED käytön. Kuitenkin ennen aloittamista laitan Node-RED palvelun käynnistymään automaattisesti käynnistyksen yhteydessä komennolla: ”***sudo systemctl enable nodered.service***”.

Tämän jälkeen voimme käynnistää Node-RED sovelluksen komenolla: ”***node-red-start***”. Ohjelma käy läpi mitä palveluita ja hakemistoja se käyttää käynnistyessään. Node-RED

ilmoittaa myös turvallisuudesta ja ehdottaa muutosta tiedostoon ”/home/pi/.node-red/settings.js”, johon siirrymme seuraavaksi. Kuitenkin sammutan toistaiseksi Node-RED palvelun komennolla: ”***node-red-stop***”, jotta voin toteuttaa turvallisuuteen liittyvät muutokset.

3.5.1 Node-RED HTTPS-suojaus

Node-RED ei oletuksena ole kovinkaan turvallinen, kuten aiemmin mainitsin. Ensimmäisenä itselleni tuli vastaa yliviivattu lukkokuvake (kuva 9), tällöin yhteyteni ei ole salattu. Alussa tuo ilmoitus ei tee mitään vaarallista, mutta myöhemmin kirjautumiset, tietokantojen hallinta ja datan kerääminen voi altistua urkkimiselle.



Kuva 9. Node-RED ilman HTTPS-suojausta.

HTTPS suojautumiseen emme nyt tutustu tarkemmin, mutta halusin kuitenkin tuoda esille haavoittuvuuden protokollassa. HTTPS suojaus vaatii huomattavasti työtä, joka itsenään voisi olla oman opinnäytetyön aihe. Lyhyesti, HTTPS vaati oman domainin, joka on autentikoitu ja jolle luodaan oma salausavain HTTPS yhteyttä varten.

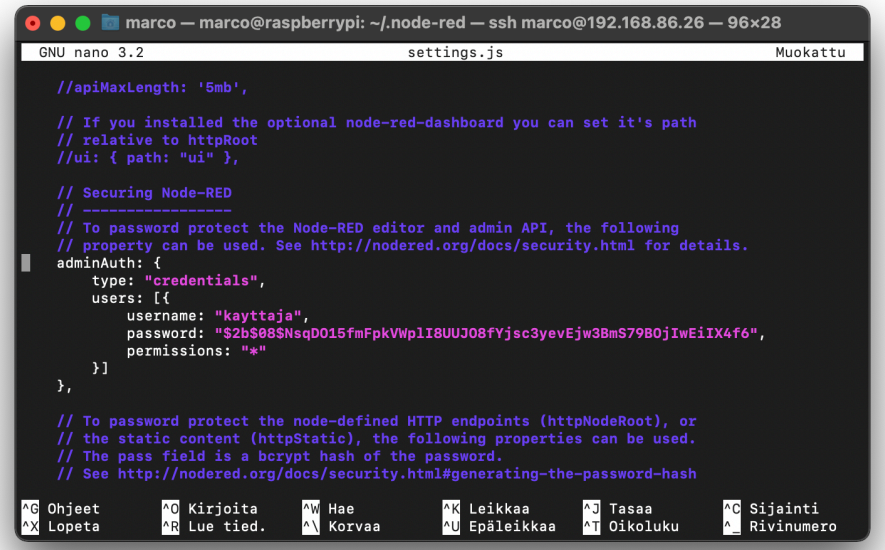
3.5.2 Node-RED vuoeditorille salasana

Tässä kappaleessa asetamme salasanan vuoeditorille. Olemme jo palomuurilla suojanneet ulkoisen pääsyn laitteelle ja nyt suojaamme vuoeditorin verkonsisäiseltä hyökkäykseltä. Kirjautuminen tapahtuu käyttäjätunnuksen ja salasanan avulla, mutta halutessaan voi ottaa jopa ulkopuolisen kirjautumistavan käyttöön esimerkiksi kirjautumisen Twitter- tai GitHub-tilin kautta.

Voit nyt halutessasi käyttää etätyöpöytä- tai SSH-yhteyttä. Sammutetaan tilapäisesti Node-RED komennolla: ”***node-red-stop***”, koska muokkaamme sovelluksen asetuksia.

Navigoi kansioon ”~/.node-red”, joko komennolla: ”**cd ~/.node-red**” tai laittamalla ”tiedoston hallinta”-sovelluksen otsikkokenttään ” ~/.node-red”.

Avaa hakemistosta ”settings.js”-tiedosto muokkaamista varten, SSH-yhteydellä voit kirjoittaa: ”nano settings.js”. Tiedosto on melko suuri ja sisältää paljon dataa, mutta meitä kiinnostaa kohta ”adminAuth:”.



Kuva 10. SSH-yhteydellä settings.js avattu Nano editorilla.

Kuvassa (kuva 10) näkyy nano editorilla avattu ”settings.js”-tiedosto. Kuvasta näkyy kohdat ”type: credentials” eli kirjautumistiedot, jonka jälkeen listataan käyttäjät. Tässä tapauksessa meille riittää yksi käyttäjä, jolla on kaikki oikeudet. Olen lisännyt ”kayttaja” kohtaan ”username:” ja salasanan kohtaan ”password”. Salasana on salattuna, joten se pitää itse generoida. Node-RED onneksi antaa meille työkalun salasanan luomista varten.

Salasanan voit generoida komennolla: ”***node-red admin hash-pw***”. Ajettuasi komennon sinulta pyydetään salasanaa, mutta se ei tarkoita tietokoneen salasanaa vaan uutta salasanaa Node-RED:iä varten. Komento siis ottaa syöttämäsi salasanan ja antaa takaisin Bcrypt salauksella tehdyn vastineen kyseiselle salasanalle. Tätä numeroiden, merkkien ja kirjaimien yhdistelmää käytetään settings.js tiedoston ”password” kohtaan. Settings.js on selkokielinen tiedosto, jolloin salasanat tulee olla salattuja.

Kun olet tallentanut käyttäjän ja salasanan ”settings.js”-tiedostoon voit käynnistää Node-RED sovelluksen uudestaan. Kun käynnistät seuraavan kerran Node-RED työpöytää

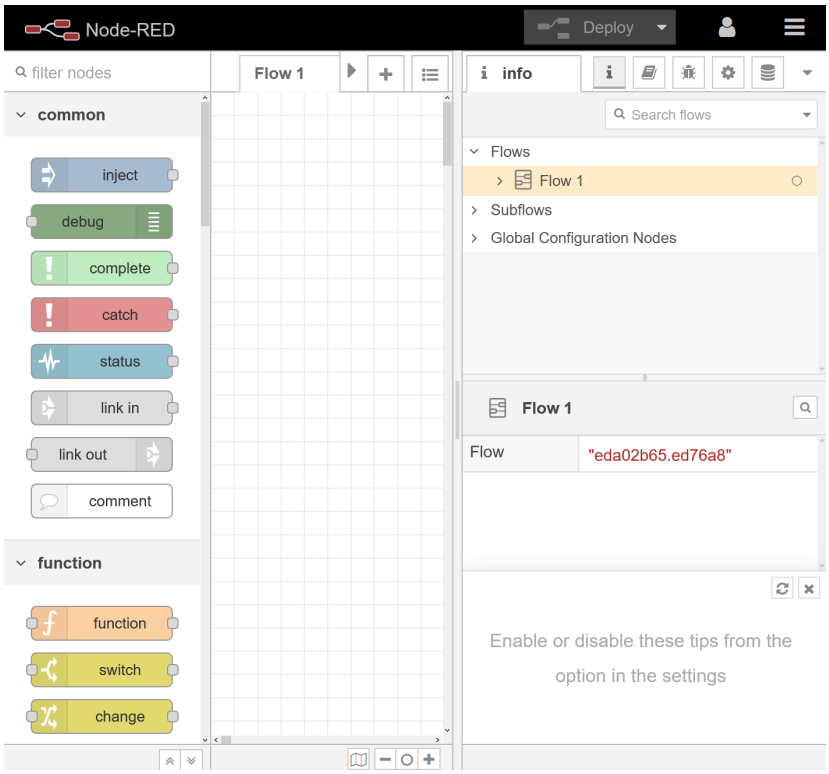
verkkoselaimessa, huomaat sen nyt vaativan sinulta salasanaa. Nyt Node-RED työpöytä on myös turvattu lähiverkon tunkeutumiselta.

3.6 Node-RED:in käyttö Raspberry PI tietokoneessa

Seuraavaksi käymme läpi Node-RED sovelluksen vuoeditorin ja sillä ohjelman toteuttamisen. Ohjelman toteuttaminen käydään vaihevaiheelta läpi, jolloin helppo pysyä mukana.

3.6.1 Node-RED vuoeditori

Avaamme nyt Node-RED vuoeditorin, jolloin pääsemme tekemään ensimmäisen ohjelmamme. Jos olet asentanut vuoeditorin Raspberry Pi tietokoneelle ja työskentelet jollain muulla laitteella, silloin sinun tulee tarkistaa Raspberryn IP osoite seuraavalla komennolla: ”***hostname -I***”. Huomaa kyseessä on iso ”i”-kirjain. Kirjoittamalla Raspberryn IP-osoitteen selaimeen ja lisäämällä perään portin: 1880, pääsemme selaimen kautta vuoeditoriin. Minulla tämä osoite on: **192.168.86.26:1880**. Jos työskentelet Raspberryllä tai olet asentanut Node-RED sovelluksen tietokoneellesi, voit avata osoitteen: **localhost:1880**.



Kuva 11. Node-RED työpöytä alussa.

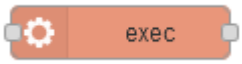
Kuvassa (kuva 11) on suurennettuna työpöytä havainnointia varten. Vasemmalla esillä on paletti, jossa on nodet listattu päällekkäin. Nodeja vedetään hiirellä keskellä olevaan työtilaan ja ketjutetaan langoilla toisiinsa. Oikealla on inforuutu, josta voi havaita tietoa ohjelman toiminnasta, eri nodejen käytöstä ympäristössä, asetukset ja ehkä alussa tärkein on debug ruutu. Debug ruutua tulemme käyttämään paljon, koska sen avulla voimme nähdä

ohjelman toiminnan tai toimimattomuuden kesken ohjelmoinnin. Kuva 3 (s. 6) demonstroi hyvin, miten nodet toimivat Debug ruudun kanssa.

3.7 Ensimmäinen ohjelma Raspberrylle

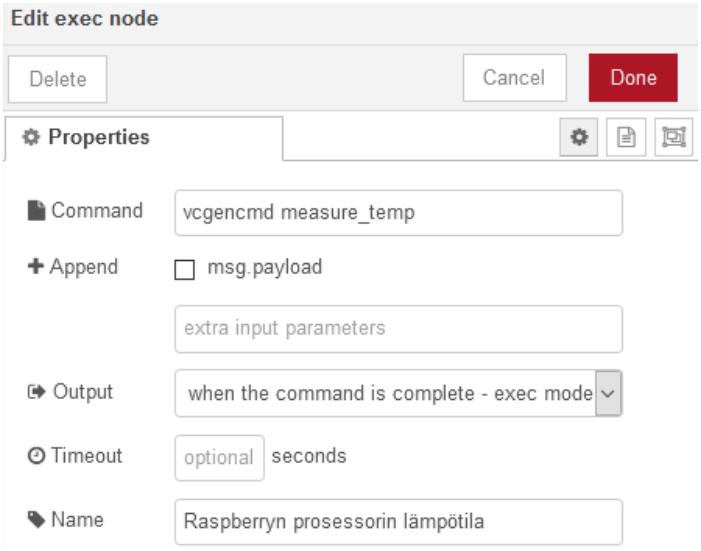
Tämä ohjelma toimii, jos Node-RED pyörii Raspberry Pi tietokoneella. Käytämme laitteen omaa järjestelmää ja ajamme sinne komennon. Raspberryn prosessorin lämpötilaa voi olla hyvä seurata, jos laite on ylikellotettuna, jatkuvassa rasituksessa tai suljetussa tilassa. Liian suuri lämpötila voi tehdä järjestelmästä epävakaan ja voi lyhentää prosessorin elinikää. Komennolla: ”***vcgencmd measure_temp***”, saamme prosessorin lämpötilan tulosteena.

Nyt otamme tuon prosessorin lämpötila komennon ja lisäämme sen Node-RED työympäristöön exec-nodella. Exec-noden oma ohje kertoo noden ajavan järjestelmäkomennon ja palauttavan sieltä tulleen tulosteen.



Kuva 12. Exec-node.

Tupla klikkaamalla nodea voimme tarkistella sen toimintaa tarkemmin. Ylimmällä rivillä näemme ”Command”-ruudun, johon voimme syöttää komentomme: ”***vcgencmd measure_temp***”. Voimme ottaa ”Append”-kohdasta ruksin pois, koska emme tuo komentoon omia lisäargumentteja. ”Name”- kohdan voit täyttää vapaasti, suosittelen toimintoa kuvaavaa termiä. Voit tarkistaa nodeen tekemäni muokkaukset kuvasta 13.



Kuva 13. Exec-noden muokkaaminen.

Muista painaa aina deploy-painiketta oikeassa yläkulmassa, kun haluat toteuttaa muutokset ohjelmassa. Ohjelma tallentuu aina kun painat deploy- painiketta.
Lisäämme debug-noden exec noden oikealle puolelle, toiminnot tapahtuvat vasemmalta oikealle Node-RED:ssä. Jolloin voimme nähdä mitä exec tulostaa meille. Ohjelma on nyt valmis, mutta se ei tiedä milloin se ajetaan. Voimme laittaa ajastimen, kellon tai vaikka meidän tapauksessamme inject-noden ajamaan ohjelman. Inject-node käynnistää exec-noden ja debug-node tulostaa exec-noden tulosteen. Ohjelma voi vaikuttaa monimutkaiselta, mutta kuvan (kuva 13) avulla voit huomata, ettei se näin ole. Inject-node oletusarvoisesti lähettää aikaleiman, mutta emme huomioi sitä exec-nodessa (huomaa ruksi, jonka poistimme aiemmin).

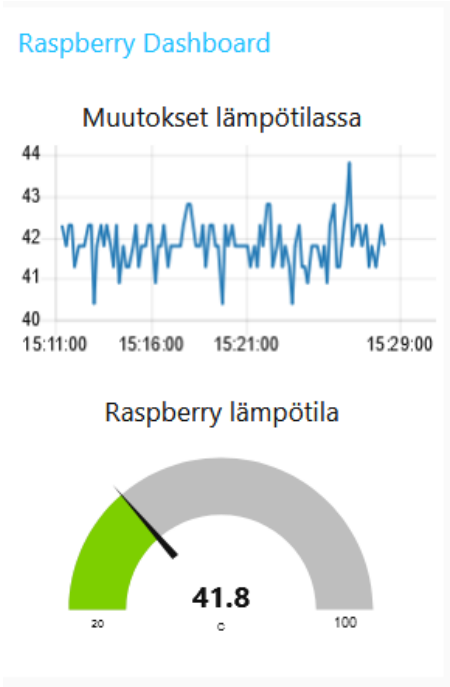
Kuva 14. Yksinkertaisen komennon ajaminen Node-RED:ssä.

Ohjelma nyt tulostaa Raspberryn prosessorin lämpötilan inject-noden vasemmalla olevaa laatikkoa painettaessa. Debug ikkunaan tulostuu string[12], eli 12 merkkiä pitkä string muuttuja, jossa lukee esimerkiksi ”temp=41.8'C”. Tuplaklikkaamalla inject-nodea voimme laittaa automaattisen injektoinnin päälle, koska avautuvan ikkunan alareunassa löytyy repeat-kohta. Repeat-kohta tarjoaa meille monta eri toisti mahdollisuutta, kuten tietyn väliajoin, tiettyyn kellon aikaan tai tietyn ajan sisällä. Voimme määrittää siis ohjelman ajamaan itsensä 10 sekunnin välein jatkuvasti tai työpäivän ajan arkisin klo 9–17.

Jos sinulla ei ole Node-RED Raspberryssä, mutta haluat kokeilla silti tätä ohjelmaa niin voit ottaa kuvasta 3 JavaScript-koodin ja laittaa sen function-nodeen. Ohjelma tällöin tulostaa päivämäärän ja kellonajan, kun painat inject-noden painiketta.

3.8 Ensimmäisen ohjelman kehittäminen pidemmälle

Aloitetaan perinteisesti kertomalla mitä haluamme saavuttaa seuraavalla ohjelmalla. Seuraavassa kuvassa (kuva 15), havainnollistetaan mihin pyrimme tässä osiossa. Tämä voi vaikuttaa hieman uhkaavan vaikealta, mutta onneksi Node-RED on hyvä työkalu tämän kaltaisia projekteja varten.



Kuva 15. Raspberry dashboard UI käytössä.

Otamme nyt ensimmäisen ohjelman ja mietimme miten voimme käyttää sitä hyödyksemme tässä projektissa. Haluammeko seurata loputonta määrä rivejä Node-RED debug-ikkunaa vai otammeko tiedot talteen ja seuraamme vain tärkeimpiä muutoksia lämpötiloissa. Visuaalisesti tekstidatan seuraaminen on melko työlästä ja eikä meidän tarvitse nähdä kaikkea tietoa mitä ohjelma vastaanottaa.

Ensimmäiseksi otamme syötteen ja nappaamme siitä vain numeron talteen, numeromuuttujana. String-muodossa voisimme vain seurata tekstikentän kirjaimien määrää, joka ei kuvasta prosessorin lämpötilaa kovinkaan hyvin. Numeromuuttujaa voimme seurata arvoina taulukossa tai graafisessa käyttöliittymässä, joten nappaamme string-muuttujasta numeron ja muutamme sen integer-numeroksi. Tässä vaiheessa huomaat, että ohjelmointia tarvitaan monimutkaisemmissa ohjelmissa. Voit hakea internetistä lisätietoa interger ja string muuttujista, mutta se ei ole nyt välttämätöntä.

Seuraavaksi lisäämme ohjelmaan function-moduulin, jonka tehtäväksi annamme halutun tiedon keruun exec-noden syötteestä. Viestit kulkeutuvat yleensä moduulien välillä viesteinä (msg.payload) ja otsikkoina (msg.topic), jolloin exec-syöte eli lämpötila tulee funktiolle muodossa ”msg.payload” (Suom. hyötykuorma).

Kuvassa 16 voit tarkastella mitä koodia olemme nyt kirjoittamassa. Otamme msg.payload syötteen ja nimeämme sen str-muuttujaksi. Str-muuttajasta nappaamme vain osan substring funktiolla ja palautamme sen msg.payload muuttujaan. Periaatteessa leikkaamme haluamamme osan pois string-muuttujasta ja tallennamme vain sen osan msg.payloadiin.

Kun olemme leikanneet osan string muuttujasta pois suosittelen testaamaan sen toimintaa debug-noden avulla. Kytke debug-node nyt function-nodeen. Debug-ikkunaan pitäisi ilmestyä vain string, jossa lukee esimerkiksi ”41.0”. Jos lukee jotain muuta voi olla, ettei substring osunut kohdilleen. Tärkeää on napata vain luku, ei mitään muuta, jotta voimme muuttaa sen numeroksi yksinkertaisella Number-funktiolla.

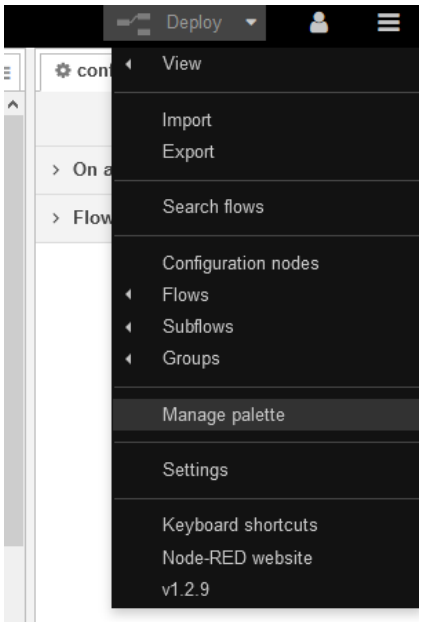
```
str = msg.payload;
msg.payload = Number(str.substring(5,9));
return msg;
```

Kuva 16. Node-RED sovelluksessa JavaScriptin subtring-komento.

Kuten kuvassa (kuva 16) näkyy, ohjelmakoodi on melko lyhyt ja yksinkertainen. Debug-ikkunassa pitäisi nyt näkyä lukuarvona prosessorin lämpötila. Ohjelmoinnissa on hyvin tärkeä huolehtia, että muuttujat pysyvät oikeina.

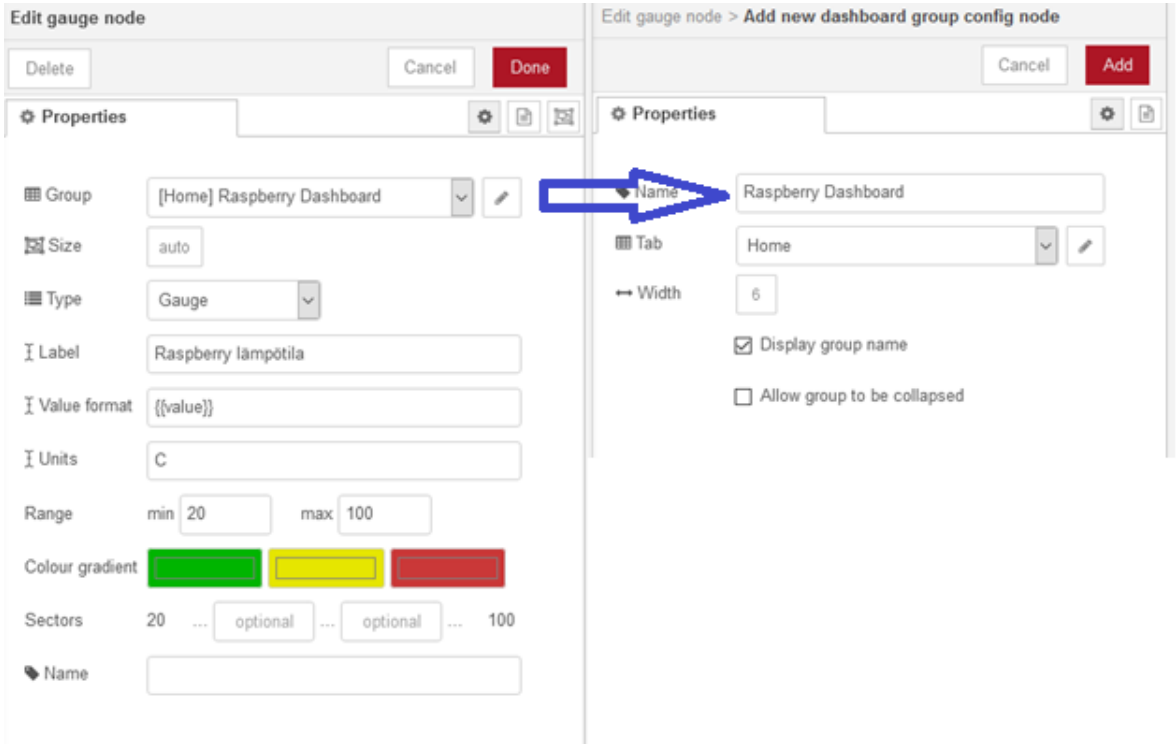
Olemme nyt saaneet numeron irti exec-komennosta ja voimme edetä visuaalisen puolen kehittämiseen. Sitä varten lataamme uusia nodeja palettiimme. Nodejen asennus tapahtuu Raspberryssä, mutta voimme ne helposti asentaa Node-RED työpöydän kautta.

Painamalla oikeassa yläreunassa olevaa kolmea viivaa aukeaa meille seuraava valikko (kuva 17). Valikossa meitä kiinnostaa ”Manage palette”-osio, koska sitä kautta voimme hakea ja asentaa meille sopivat nodet. Valittuasi ”Manage palette”-osion auki, avaa ”Install”-välilehti ja kirjota sinne: ”dashboard”. Etsi listalta ”node-red-dashboard” ja paina sen perässä olevaa install-painiketta.



Kuva 17. Node-RED valikko.

Dashboard asentaa useita eri nodeja käyttöömme, joista otamme nyt ensin gauge-noden käyttöön. Gauge-nodella saamme verkkosivulle nopeusmittarin seuraamaan prosessorin lämpötilaa. Aloita laittamalla gauge-node työtilaan ja liitä siihen funktio-node. Seuraavaksi tupla klikkaa gauge-nodea ja ensimmäiseksi muokkaa ”Group”-kohtaa, painamalla kynän ikonia. Anna ryhmälle vain nimi ja paina Add. Tämä luo pohjan, johon tulevat dashboardin nodet sijoittuvat. Kun olet takaisin gauge-noden asetuksissa voit muokata kohdat: ”Label”, ”Units” ja ”Range” kuvan 18 mukaisesti.



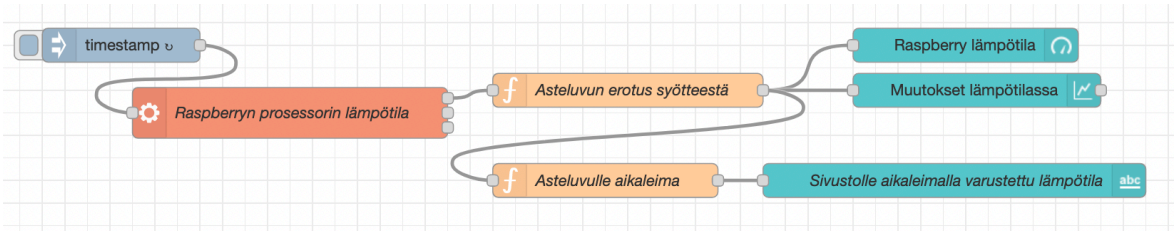
Kuva 18. Dashboardin luominen ja gauge-node.

Painettuasi deploy voit navigoida omalle verkkosivulle, joka löytyy lisäämällä ”/ui” Node-RED osoitteen perään. Esimerkiksi: ”**localhost:1880/ui**”. Nyt sinun pitäisi nähdä kuinka lämpötila vaikuttaa mittarin toimintaan. Jos mittari ei liiku tai rekisteröi mittauksia, voit tarkistaa vielä inject-noden toiminnan ja laittaa sen esimerkiksi toistamaan injektiot 10 sekunnin välein.

Seuraavaksi toteuta sama chart-nodella, vaiheet ovat melkein samat kuin gauge-nodessa, mutta nyt ei tarvitse luoda dashboardille ryhmää vaan sen pitäisi valmiiksi näkyä jo nodessa. Muista jälleen painaa deploy ja lataa ”ui”-verkkosivu uudestaan. Sitten näkymän pitäisi muistuttaa kuvan 15 näkymää.

Mitä iloa on seurata prosessorin lämpötilaa näin verkon kautta ja tehdä viivakaavio siitä? Tällä samalla tekniikalla voit tarkkailla kymmenien tietokoneiden toimintaa ja lämpötiloja. Miksi rajoittaa pelkkiin tietokoneisiin, kun voisimme myös liittää antureita Raspberryyn tai lähettää siihen bluetoothin kautta anturidataa kymmenistä eri mittareista. Voimme seurata, vaikka kaupan kaikkien kylmäkaappien lämpötiloja tai kodin kasvien mullan kosteutta yhdestä ruudusta. Yhdistämällä esimerkiksi Arduinon tai ESP32 moduulin voimme luoda seuranta dataa melkein mistä vain ja Node-RED avulla sen toteuttaminen vaatii vain muutaman node-palikan ja hieman JavaScript-koodia.

Lisäsin tähän ohjelmaan vielä funktion, joka ottaa lämpötilan vastaan ja laittaa sille päivämäärän ja kellonajan. Jolloin voimme seurata mitä on mitattu ja milloin. Tämä tieto voidaan tallentaa reaaliajassa tietokantaan tai lokitiedostoon, jolloin on tärkeää mainita ajankohdat eri merkinnöille.



Kuva 19. Lopullinen ohjelma Node-RED työpöydällä.

3.9 MQTT Node-RED sovellus

Seuraavaksi teemme sovelluksen, joka on hieman monimutkaisempi ja vaatii perehtymistä asiaan. Aiheen ollessa melko laaja voimme tehdä pienemmän sovelluksen, jolloin tekniikka on painopisteenä eikä ohjelman laajuus.

3.9.1 HSL MQTT-broker

MQTT, kuten aiemmin (kappale 2.3) mainitsin, on tehokas kommunikaatio keino etenkin kahden laitteen välille (M2M eli machine to machine). Helsingin Seudun Liikenne (HSL) käyttää MQTT teknologiaa liikenteen seuraamiseen ja sen avulla on toteutettu runsaasti sovelluksia eri alustoille, jotka osaavat hyödyntää kyseistä kommunikaatiota. Puhelimissa voidaan sovellusten avulla seurata reaaliajassa bussin, raitiovaunun tai junan liikkumista kartalla. Älykkäiden algoritmien avulla voidaan myös ennustaa koska kyseinen julkinen kulkuneuvo saapuu kohdallesi. HSL liikenne tietenkin noudattaa aikatauluja, mutta mielestäni on hyvä tietää missä kyseinen kulkuneuvo on menossa.

HSL on tehnyt MQTT protokollastaan julkisen, jolloin käyttäjät voivat toteuttaa omat ohjelmansa sen avulla. Toteutamme seuraavaksi MQTT protokollan avulla Node-RED sovelluksen. Ensiksi meidän pitää käydä Digitransit MQTT sivustolla, jotta saamme MQTT Brokerin osoitteen. Broker ottaa MQTT-dataa vastaa ja toimittaa sen eteenpäin tilaajille eli toimii tiedon välittäjänä. Tietojen tilaaminen MQTT-protokollassa tapahtuu verkko-osoitteen perusteella.

Digitransit sivusto tarjoaa meille lukuisia eri API end-pointteja, mutta etsimme salamattoman ja kevyen MQTT endpointin. Tämä endpoint on riittävä meidän esimerkkimme tapaukseen, mutta sovelluksiin tulisi käyttää salattua yhteyttä. Digitransit suosittelee TLS-yhteyttä, jotta käyttäjien tiedot pysyisivät yksityisinä (Digitransit 2021).

Digitransit sivustolla tällä hetkellä toimiva MQTT-osoite on: ”**mqtt://mqtt.hsl.fi:1883/**”. Mutta suosittelen tarkistamaan kyseisen osoitteen Digitransit palvelusta ennen käyttöä. Kyseinen MQTT-osoite kertoo meille nyt MQTT-brokerin ja käytössä olevan portin, mutta tiedon saaminen vaatii vielä lisää argumentteja osoitteeseen. Argumentin avulla voidaan tilata ”topic” eli aihe, josta haluamme vastaanottaa tietoja. Oikean argumentin rakentaminen on ehkä työläin osuus tämän ohjelman teossa, mutta käyn läpi sen rakentamisen.

Esimerkki osoite sivustolla (Digitransit 2021) näyttää meille miten kokonainen argumentti rakentuu eli miten tietyn ajoneuvon tietojen tilaaminen onnistuu. Esimerkkinä on seuraava osoite: “**/hfp/v2/journey/ongoing/vp/bus/0055/01216/1069/1/Malmi/07:20/1130106/2/60;24/19/73/44**”. Nopeasti lukemalla huomataan kyseessä olevan bussi Malmilla ja kello 7:20, mutta muut tiedot eivät kerro juuri mitään. Tämä on kuitenkin täysin toimiva komento oikean aiheluokan tilaamiseen.

Digitransit sivusto (Digitransit 2021) selittää meille, kuinka argumenttia luetaan:
”/hfp/” – Aihepuun juuri.
”/v2/” – HFP (High-frequency positioning) versio.
”/journey” – Kulkuneuvon on ajossa.
”/ongoing” – Kulkuneuvo on ajamassa reittiään.
”/vp” – Kulkuneuvon sijainti.
”/bus” – Kulkuneuvotyyppi.
”/0055” – Kulkuneuvon omistava yritys.
”/01216” – Kulkuneuvon tunnistenumero.
”/1069” – Reitin tunnistenumero.
”/1” – Kulkuneuvon menosuunta reitillä.
”/Malmi” – Kulkuneuvon kyltin teksti.
”/7:20” – Kulkuneuvon ajonaloitus.
”/1130106” – Seuraava pysähdys.
”/2” – Geohash tarkkuus (eli kuinka tarkkaan sijainti päivittyy).
”60;24/19/73/44” – Geohash eli sijainti.

Lyhyesti kerrottuna: “**/hfp/v2/journey/ongoing/vp**” – aloittaa argumentin ja haluamme tietää kulkuneuvon sijainnin. Seuraavaksi tulevat:
/kulkuneuvo
/yritys
/kulkuneuvon tunniste
/reitti
/kulkusuunta
/kyltin teksti
/lähtöaika
/seuraava pysähdys
/geohash tarkkuus
/geohash

Halutessasi voit jättää myös kohtia tyhjäksi, kuten yrityksen tunnuksen, tällöin voidaan laittaa korvausmerkki kyseiseen luokkaan ”**+**”-merkillä. Emme välttämättä tiedä mitä kyltissä lukee ja mikä yritys kyseistä linjaa ajaa, jolloin korvausmerkit ovat tällöin käytännöllisiä. Jos haluamme lopettaa argumentin kesken ja saada tiedot kaikista kulkuneuvoista, jotka ovat aiempien argumenttien parametrien sisällä, voimme laittaa ”#”-merkin korvausmerkiksi.

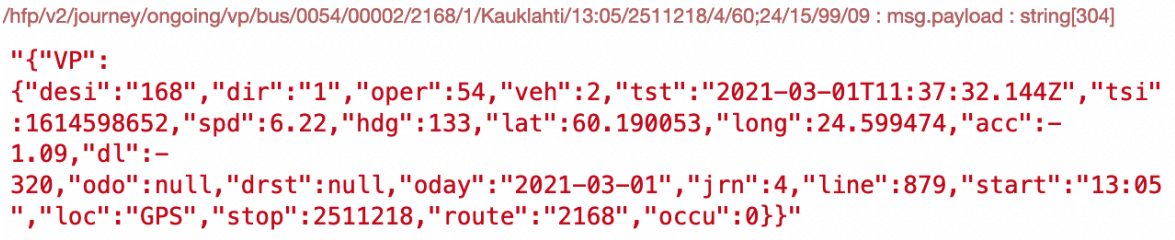
Korvausmerkkien käyttäminen tulee tehdä harkitusti, jos laitat pelkän alkuosan ja päätät sen ”#”-merkillä saat kaikki liikennetiedot mitä HSL lähettää. Esimerkkinä vielä kyseinen argumentti: ”**/hfp/v2/journey/ongoing/vp/#**”. Palvelu lähettää kaikki tiedot HSL liikenteestä tietokoneellesi ja se tekee parhaansa niiden tulkitsemiseen, mutta mitään käytännön hyötyä ei ole tämänkaltaiselle tietotulvalle. Pahimmassa tapauksessa ohjelma tai sen tulkki saattaa kaatua tai mennä jumiin.

Korvausmerkkejä voi käyttää seuraavalla tavalla: “/**hfp/v2/journey/ongoing/vp/+/+/+/+/+/Kauklahti/#**”. Kyseinen komento haluaa vain tietää kulkuneuvon, jonka kyltissä lukee Kauklahti. Kyseinen komento tuottaa myös hyvin paljon tietoa, koska Kauklahti on monen junan ja bussin päätepiste. Huomaa kuinka ohitimme ensimmäiset viisi kysymystä ”**/+/+/+/+/+/**” ja Kauklahden jälkeen ”**#**”-merkki ilmoittaa, ettei meitä kiinnosta kulkuneuvon tämänhetkinen sijainti. Jos laitamme geohash koordinaatit pyyntöön, voimme rajaa seurannan tietylle alueelle.

Seuraavaksi voimme käydä läpi Geohash paikannuksen HSL sovellusta varten. Geohash osoitteen muodostaminen on melko yksinkertaista. Aiemman esimerkin 60;24/19/73/44 geohash on GPS muodossa lat 60.174 long 24.934. Tästä huomaa nopeasti kaavan, jossa pisteen jälkeiset luvut ovat jaettuna lokeroihin pareittain. Esimerkiksi lat 11.111 ja long 22.222 muuttuvat 11;22/12/12/12 geohashiksi. Ensimmäiset kaksi lukua laitetaan ensimmäiseen lokeroon ja erotetaan toisistaan puolipisteellä, seuraaviin lokeroihin laitetaan vuorotelleen sijainnin tarkemmat luvun vuorotellen pareittain.

3.9.2 HSL MQTT hyötykuorma

Hyötykuorma eli payload, tarkoittaa sitä tietoa, jonka tilaamme MQTT-brokerilta edellisen kappaleen ohjeiden perusteella. Hyötykuorma sisältää argumenttiin liittyviä tietoja ja sen lisäksi reaaliaikaista dataa kulkuneuvosta.



Kuva 20. HSL MQTT-payload Node-RED:ssä.

Kuvassa (kuva 20), näemme mitä tietoa yksittäinen kulkuneuvo lähettää MQTT-brokerille ja sen jälkeen meille.

Digitransit (2021) sivuston avulla voimme tarkastella meille tärkeitä tietoja hyötykuormasta:
”desi”- Asiakkaille näkyvä reitintunniste.
”dir”- Reitinsuunta.
”spd”- Nopeus.
”hdg”- Suunta.
”lat/long”- Sijainti (latitude/longitude).
”start”- Reitin lähtöaika.
”stop”- Pysäkin tunnus.
”route”- Reitin tunnus.

Tarkempia tietoja voit käydä lukemassa Digitransitin verkkosivustolta.

HSL ei vielä hyödynnä kaikkia mahdollisia tietolähteitä hyötykuormassaan, mutta on selkeästi valmiina niitä käyttämään. Hyvänä esimerkkinä on ”occu” eli matkustajat, jolloin kulkuneuvon tietoihin kirjautuu matkustajien määrä. Matkustajien määrää seurataan tällä hetkellä vain Suomenlinnan lauttareiteillä, mutta voi tulevaisuudessa ilmoittaa eri linjojen ruuhkatilannetta.

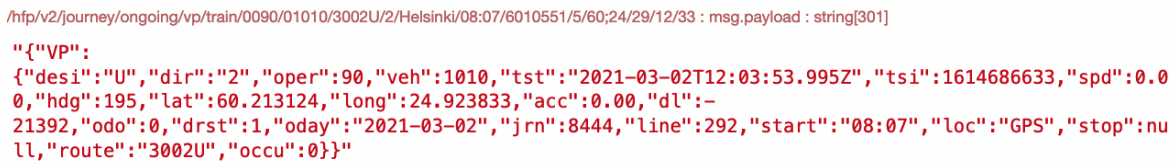
Tästä hyötykuormasta voimme napata tietyn tai tiettyjen kulkuneuvojen tiedot. Voimme seurata esimerkiksi bussien nopeuksia tietyllä alueella tai seurata yksittäisen bussin kulkua.

3.9.3 Node-RED hyödyntämässä HSL MQTT-palvelua

Edellisten kappaleiden selostukset ja esimerkit ovat nyt avartaneet käsitystämme MQTT-protokollasta, joten voimme siirtyä sen käyttämiseen Node-RED:ssä. Node-RED sisältää valmiiksi MQTT noden, johon lisätään MQTT-brokerin tiedot ja seurattavan MQTT-aiheen. MQTT-Brokerin osoite on ”**mqtt://mqtt.hsl.fi**” ja portti **1883**, kirjautumistietoja ei tarvitse.

Seuraavaksi voimme miettiä mitä haluamme tehdä MQTT tiedoilla. Voimme seurata kulkuneuvojen liikennettä GPS sijainnin perusteella, joten varmaan voimme seurata esimerkiksi junien liikennettä kartalla. Käydään ensin läpi mitä tietoa tarvitsemme ja kuinka tietoa voidaan kalastella MQTT viestistä. Joten laitamme nyt MQTT-noden tekemään laajan kyselyn, koska voimme helposti napata vastauksesta meille sopivan argumentin seuraavaa kyselyä varten. Voisimme kokeilemalla saada myös oikean argumentin, mutta aikaa voidaan säästää nopealla laajalla kyselyllä.

MQTT-nodeen lisäämällä “topic”-kohtaan “**/hfp/v2/journey/ongoing/vp/train/#**”, saamme tietää kaikkien paikallisjunien tiedot HSL alueella. Debug-noden voi lisätä ohjelmaan valmiiksi pois päältä kytkettynä painamalla noden oikealla olevaa laatikkoa kerran. Painettuasi deploy-painiketta ei tapahdu vielä mitään ja debug-ikkuna on edelleen tyhjä. Testasin itse kyseistä toimintoa keskellä päivää ja aktivoituani debug-noden sain noin kahdeksankymmentä MQTT viestiä kahden sekunnin aikana. Kyseessähän on koko pääkaupunkiseudun junaliikenteen reaaliaikaiset tiedot ja tuossa ajassa voi saada samasta junasta useamman ilmoituksen. Huomaa myös, että junassa voi olla useampi veturi, joka ilmoittaa tietonsa MQTT-palveluun.



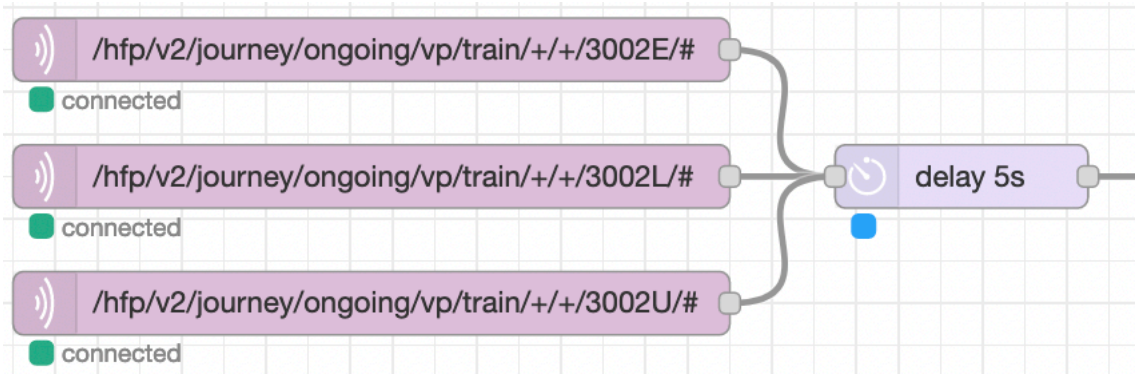
Kuva 21. MQTT-viesti U-junalta.

Löysin haluamani tiedon hyötykuormasta helposti etsimällä **”desi”:”U”** kohdan hyötykuormasta (kuva 21). Tiedon olisi toki voinut löytää erillisellä funktiolla tai tarkemmalla argumentilla, mutta tällä kertaa näin on helpompi löytää se mitä etsii. Jos teet haun kaikista lentokoneista Suomen yläpuolella, tällöin tarkempi funktio tai argumentti tekee tiedon selaamisesta helpompaa ja joissain tapauksessa mahdollista. Liian suuri tiedon määrä kerralla voi jumittaa tietokoneen tai Node-RED sovelluksen. Tietoa tulee tässä ohjelmassa myös paljon, joten hidastin sen toimintaa delay-nodella, mutta palaamme siihen myöhemmin tässä osiossa.

MQTT-topic tiedosta voimme huomata seuraavan: ”**/3002U/2/Helsinki/**” eli reitti 3002U, suunta 2 ja kyltissä lukee Helsinki. Voimme siis päätellä 3002 on rataosuuden koodi ja sen perässä on junan kirjain. Suunta 2 tarkoittaa Helsingin suuntaan, koska kyltissä lukee näin. Hyötykuormasta löydämme myös suunnan 2 (**”dir”:”2”**), mutta kyltin tiedot puuttuvat hyötykuormasta.

Voimme siis tehdä uuden MQTT argumentin ja otamme vain ne tiedot mitkä meitä kiinnostaa: ”**/hfp/v2/journey/ongoing/vp/train/+/+/3002U/#**”. Muutokset tehtyäsi voit painaa jälleen deploy ja tarkistaa mitä debug-ikkunaan ilmestyy, muista kytkeä debug-node hetkeksi päälle. Tietoa tulee jälleen huomattavat määrät, mutta ei lainkaan niin paljoa kuin aiemmin, koska nyt saamme tiedot vain U-junan liikenteestä Helsinki-Kirkkonummi-Helsinki välillä. Valitsin esimerkkinä Espoon junat, koska niitä käytin pääasiassa opiskelun aikana.

Nyt olemme eritelleen yhden linjan tiedot eli tiedämme missä U-junat liikkuvat. Haluan myös tietää E-junan ja L-junan tiedot, koska ne liikkuvat myös samalla Espoon radalla. Voimme siis tehdä argumentit kyseisille linjoille uusiin MQTT-nodeihin laittamalla topic kohtaan ”**/hfp/v2/journey/ongoing/vp/train/+/+/3002L/#**” ja ”**/hfp/v2/journey/ongoing/vp/train/+/+/3002E/#**”. Nyt meillä on kolme eri MQTT-nodea, joista saamme kyseisten linjojen liikenteen. Tiedon määrä on nyt melkoinen, joten voimme liittää MQTT-nodet suoraan delay-nodeen.



Kuva 22. Delay-noden käyttö.

Kuvasta (kuva 22) voimme havaita MQTT-nodet, joiden alapuolella lukee “connected” eli MQTT-nodet ovat yhteydessä MQTT-brokeriin. Tämän jälkeen viestit kulkeutuvat delay-nodeen, joka oletusarvoisesti pysäyttää viestit viiden sekunnin ajaksi. Tämä on meille riittävä tauko, mutta se tekee ohjelmasta hieman kevyemmän ja näitä delay-nodeja voi myös laittaa muiden nodejen väliin.

Muutamme seuraavaksi saapuvan tiedon JSON muotoon, jolloin se on helpommin käytettävissä eri sovelluksille. Tämä on onneksi tehty melko helpoksi Node-RED nodejen avulla. Laittamalla JSON-noden heti delay-noden perään muutamme kaikki tulevat junatiedot JSON muotoon. JSON tiedot voit tarkistaa laittamalla debug-noden sen perään ja tarkistamalla mitä dataa sieltä tulee nyt ulos. Debug-noden avulla voimme tarkistaa miten tieto muuttuu eri nodejen välillä. Hyvin samantapainen käytäntö löytyy myös normaalissa ohjelmoinnissa, jolloin käytetään console.log funktiota kirjaamaan mahdolliset virheet tai virhekohdat ohjelmista ja tulostaa ne ajoympäristön konsoliin. Node-RED käyttää konsolin sijaan debug-ikkunaa, mutta vakavammat virheet kirjautuvat Node-RED lokeihin tai ohjelman ajoikkunaan (ajoikkuna ei ole käytössä automaattisesti, jos Node-RED toimii palveluna tietokoneella).

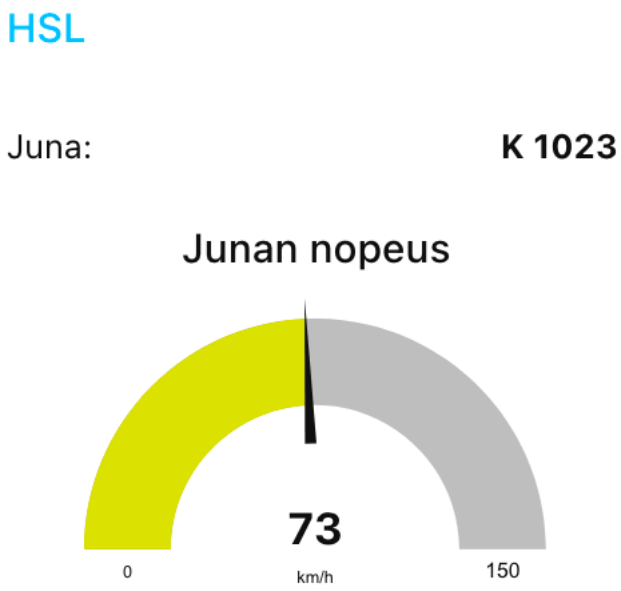
JSON tiedot siirtyvät seuraavaksi funktio-nodelle. Kartta-node, jonka kohta asennamme, vaatii tietojen olevan tietyssä muodossa. Karttasovellus vaatii jokaiselle merkille uniikin nimen. Karttasovelluksessa pitää tietenkin mainita mikä juna on kyseessä, jolloin pelkän veturin numero ei voi olla nimi. Tämä kohta jäi minua askarruttamaan pitkäksi ajaksi, kunnes tajusin tehdä nimen yhdistämällä junan kirjaimen ja veturin numeron.

```
var junakirjain = "";
var suunta = "";

if (msg.payload.VP.dir == 1) {
    suunta = "Espoo";
} else if (msg.payload.VP.dir == 2) {
    suunta = "Helsinki";
}

if (msg.payload.VP.desi == "L") {
    junakirjain = "L " + msg.payload.VP.veh + " kohti: " + suunta;
} else if (msg.payload.VP.desi == "U") {
    junakirjain = "U "  + msg.payload.VP.veh + " kohti: " + suunta;
} else if (msg.payload.VP.desi == "E") {
    junakirjain = "E " + msg.payload.VP.veh + " kohti: " + suunta;
} else {
    junakirjain = "Tuntematon juna ";
}

sijainti = { "name": junakirjain, "lat":msg.payload.VP.lat, "lon":msg.payload.VP.long,
        icon: "fa-train", iconColor:'#0000FF'};

msg.payload = sijainti;

return msg;
```

Kuva 23. Funktio junien erotteluun kartalla.

Samanimiset junat kartalla vilkkuivat monessa eri kohdassa, koska ohjelma vaan näyttää missä kyseisen kirjaimen juna liikkuu tällä hetkellä. Lisättyämme junan nimeen myös veturin numeron, joka on uniikki, saimme yksilöllisen nimen koko junalle. Kuvassa (kuva 23)

voit huomata lisänneeni nimeen myös junan suunnan, jolloin se on mielestäni informatiivisempi. Ohjelma tallentaa muodostamansa nimen hyötykuormaan ja ottaa alkuperäisestä hyötykuormasta GPS-sijainnin junalle.



Kuva 24. MQTT-kommunikaatio edellisen ohjelman koodilla.

Kuvassa (kuva 24) näet esimerkin, miten tietoa voidaan yksinkertaisesti käyttää, vaikka edellisen ohjelman pohjalta. Otin aiemman ohjelman dashboardin käyttöön ja lisäsin sinne teksti-noden junan nimeä varten ja gauge-noden junan nopeudelle. Gauge-node seuraa junan nopeutta, joten joudut tekemään funktion laskemaan muunnoksen metriä sekunnissa kilometreihin tunnissa. JSON muunnoksen jälkeen ja mukauttamalla kuvan 23, koodia voit helposti luoda samankaltaisen sivun.

Huomaamme myös kuvasta 24, että kyseessä on K-juna. Minun piti seurata Espoon junia eikä muita pääkaupunkiseudun junia, mutta tässä tapauksessa käytin veturin tunnusta seuratakseni sen nopeutta. Jos seuraan reitin 3002U junan nopeutta niin saan yhden tai useamman junan nopeuden, jolloin mittari hyppii levottomasti kahden junan nopeudesta riippuen. Tätä demonstraatiota varten käytin MQTT-aiheena: ”**/hfp/v2/journey/ongoing/vp/train/+/01023/#**” eli seurasin junanumero 1023 liikkumista. Veturit eivät jatkuvasti liikennöi samoilla reiteillä ja tässä tapauksessa seurasin U-junaa, joka seuraavana päivänä olikin K-juna.

Voimme jatkaa siitä mihin jäimme funktion jälkeen eli tarkistamme funktion toiminnan debug-nodella. Seuraavassa kuvassa (kuva 25) näemme objektin, jolla on seuraavat ominai-

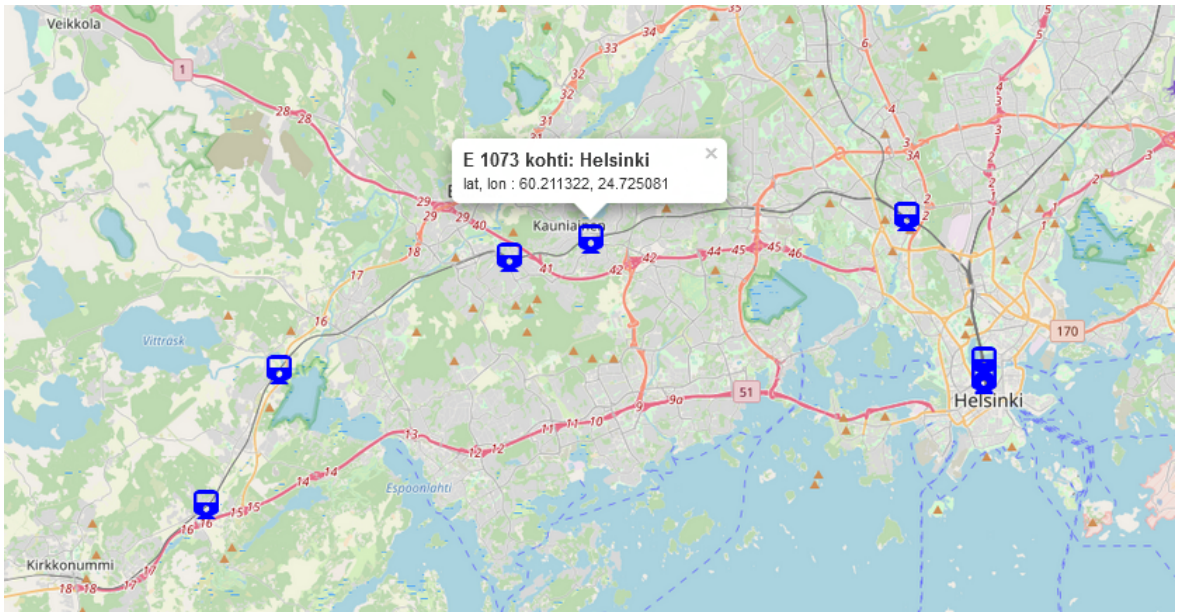
suudet nimi, latitudi, longitudi ja ekstrana vielä sininen junaikoni karttaa varten. Kartta sovellus ei vaadi erillistä ikonia, mutta minusta kartalla liikkuva juna kuvastaa paremmin mitä olemme seuraamassa.

```
{ name: "U 1030 kohti: Helsinki", lat: 60.184739, lon: 24.573964, icon: "fa-train", iconColor: "#0000FF" }
```

Kuva 25. Funktion välittämä objekti karttasovellukselle.

Kuvassa (kuva 25) voimme havaita nimen olevan nyt yksilöllinen ja GPS tiedot ovat numeerisia. Joskus tiedot saattavat olla string-muodossa, jolloin ne pitää muuttaa ennen numeroiksi ennen käyttöä. Ohjelma toimii normaalisti ja antaa oikean tulosteen niin voimme siirtyä kartta-noden asentamiseen. Eli kuten opinnäytetyön aiemmassa ohjelmassa, menemme Node-RED palettiin ja valitsemme ”Install”. Kirjoita hakukenttään: ”worldmap” ja valitse ja asenna ohjelma: ”node-red-contrib-web-worldmap”.

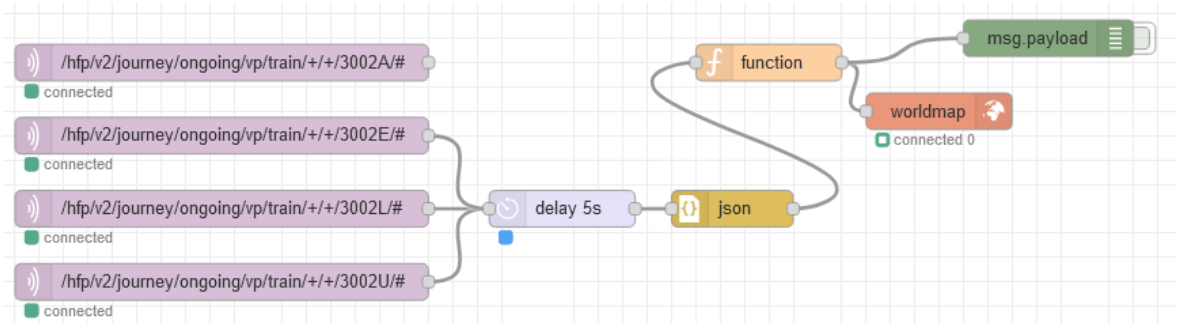
Asennettuasi worldmapin vedä worldmap-node ohjelmaasi ja kytke funktio-node siihen kiinni. Avaa worldmap-noden asetukset ja muokkaa seuraavat kohdat: Start Latitude 60.19, Longitude 24.65 ja Base map vaihda OpenStreetMap. Asetuksilla kohdistamme kartan pääkaupunkiseudulle ja karttana käytämme melko selkeää OpenStreetMap palvelua. Paina deploy, kun olet muutokset tehnyt. Tämän jälkeen siirry selaimella ”/worldmap” osoitteeseen, jos Node-RED on asennettu omalle koneellesi se löytyy osoitteesta: ”**localhost:1880/worldmap**”.



Kuva 26. Karttasovellus worldmap-noden avulla.

Karttasovelluksen pitäisi nyt avautua suoraan oikealle kohdalle ja junat ilmestyvät kartalle. Junia klikkaamalla saat nimen näkyviin. Ikoneita ja muita ominaisuuksia voit vaihtaa

worldmap-noden tekijän sivuston ohjeiden mukaisesti. Käyttämäni junan ikoni on FontAwesomen palvelusta. Ikonien käyttäminen ei vaadi erillistä asennusta.



Kuva 27. Node-RED HSL karttasovellus.

Kuvassa (kuva 27) olen laittanut junareitit päällekkäin, jolloin voin valita mitä kartalla näytetään. Loppujen lopulta melko yksinkertainen sovellus, kun ymmärtää miten MQTT toimii ja miten sen lähettämää tietoa käsitellään. Sovellusta voisi tietenkin vielä laajentaa, mutta yksinkertaisempi sovellus on helpommin ymmärrettävä. Seuraava askel voisi olla dashboardin sisällä pyörivä kartta, johon junat listautuvat ja kartalla näkyvät junat voisi valita listalta. Tämän kaltaisia ohjelmia on jo tehty montakin, mutta itse tekemällä voi oppia uusia asioita.

4 Pohdinta

Tämän opinnäytetyön avulla voi luoda oman IoT-laitteen ja tehdä sille sovelluksia. Ohjeistuksen avulla luodaan kolme eri esimerkki sovellusta, joita pystyy itsenäisesti päivittämään ja parantamaan. Kannustan jatkokehittämään, koska silloin oppii parhaiten tekniikoiden käyttöä. Mielestäni ohjeet ovat melko kattavat aiheeseen liittyen, jokaiseen kohtaan voisi syventyä lisää, mutta mielestäni on hyvä rajata tämä projekti tiiviiksi kokonaisuudeksi.

Haasteina tämän opinnäytetyön tekemisessä olivat aiheen kiinnostavuus, aiheen samankaltaisuus muiden opinnäytetöiden kanssa ja melko laajan prosessin supistaminen mahdollisimman pieneksi. Kiinnostavuutta päätin lisätä avaamalla koko projektin aloittelijan tasolle, jolloin opinnäytetyötä luettaessa oppii mistä puhutaan seuraavassa kappaleessa. Samankaltaisuuden ongelman sain myös samalla tekniikalla ratkaistua, koska lähdin aivan alusta liikkeelle enkä oleta lukijan osaavan käyttää esimerkiksi Raspberry Pi tietokonetta. Laajuutta mietin pitkään ja tämä asia olikin opinnäytetyön aloittamisen suurimpia haasteita, mutta sainkin oivalluksen tehdä kolme helpompaa ohjelmaa yhden laajemman sijasta. Yhden ohjelman tekeminen olisi vaatinut paljon perehtymistä lukijalta hyvin lyhyessä ajassa, mutta kolmen ohjelman avulla pystyin toteuttamaan perehtymisen vaiheittain.

Omaa oppimistani tuki ehdottomasti into oppia uutta Raspberry Pi tietokoneesta ja Node-RED sovelluksesta. Ohjelmien tekeminen oli onneksi jo tuttua, mutta viimeinen ohjelma toi minulle lisähaastetta, koska otin ensimmäistä kertaa käyttää worldmap laajennuksen. Jos olisin vaatinut alusta asti itseltäni karttasovelluksen, jolla seurataan junia, olisin varmaan murehtinut sitä koko opinnäytetyön ajan. Mutta annoin itselleni aikaa oppia ja pystyinkin toteuttamaan ohjelman hyvin. Karttasovelluksen ensimmäinen merkki junasta oli minulle jo suuri voitto ja seuraavaksi halusin sen jo liikkuvan kartalla. Karttamerkit hyppivät levottomasti ja välillä monistuivat moninkertaiseksi, jolloin se muistutti enemmän Nokian vanhaa matopeliä. Jokainen ongelma ja niiden ratkaiseminen toivat paljon uutta minulle.

Jatkokehittämistä voi ohjelmille tehdä loputtomasti ja itse IoT-projektia voisi laajentaa myös ulkoisten LED-lamppujen tai näyttöjen avulla. Tai jos tilastoja harrastaa voi tehdä SQL tietokannan ja tallentaa keskinopeuksia tai jarrutus/kiihdytys nopeuksia. Itse haluaisin kehittää sovellusta vielä seuraaman junia dashboardin kautta ja kartalla junat voitaisiin valita listalta. Huomaa kuitenkin miten paljon tietoliikennetietoja lataat koneellesi, koska minulla meni helposti 4-5 Mbps sovelluksen ollessa käynnissä.

Opinnäytetyötäni tehdessä opin tietenkin itse samalla uusia asioita Raspberry OS:sta ja toin oivallukset osaksi tätä projektia. Linux on myös käyttöjärjestelmä, jota en käytä päivittäin vaan yleensä eri projektien yhteydessä, joten sen käyttäminen opettaa minulle aina

uusia käytänteitä. Käyttäjätunnusten luominen ja tärkeiden komentojen turvaaminen oli minulle uutta. Turvallisuus näkökulma toi itselleni paljon uutta, mitä en aiemmin tiennyt.

Sudo-komentojen ajamisen esto Raspberryn puolelta teki laitteesta yllättävän turvallisen. Node-RED pystyy ajamaan komentoja suoraan vuoeditorista, mutta estettyämme sudo-komentojen ajamisen ilman oikeaa salasanaa tulimme estäneeksi myös haitallisten komentojen ajon ohjelmassamme. Ei kriittisiä komentoja voi ajaa ohjelmassa, kuten lämpötilan seurantaa, mutta ei voi tehdä mitään vaarallista. Pieni paha on tehdä exec-komento, joka sammuttaa Raspberryn, hieman isompi paha on tyhjentää koko tietokone komennolla: ”**sudo rm -rf /***” ja pahin olisi ehdottomasti tiedon keruu lähiverkosta.

Opinnäytetyö oli tietenkin työläs prosessi, koska aihe ei ole pelkästään kirjallisuuden tutkimista vaan myös oikean elämän projekti. Raspberry Pi tietokoneita minulla on onneksi pari kappaletta, jolloin pystyin yhden jättämään Node-RED projektiksi tämän opinnäytetyön ajaksi. Ilman Raspberry tietokonetta olisi tämä opinnäytetyö jäänyt pelkäksi Node-RED oppaaksi, joita ei suomen kielellä ole myöskään liikaa. Halusin kuitenkin pitää IoT-periaatteen osana opinnäytetyötäni. Node-RED tekee mahdolliseksi ohjelmoida milloin vain eikä sen käynnistäminen vaadi mitään vaivaa, koska Raspberry on aina käynnissä ja myös Node-RED. Syötä selaimeen Node-RED osoite ja olet jo ohjelmoimassa!

LÄHTEET

Barrett, Daniel J.; Silverman, Robert E.; Byrnes, Robert G, 2005. SSH, The Secure Shell: The Definintive Guide, 2nd Edition. O'Reilly Media Inc. USA.

Baxter-Reynolds, Matthew 2011. Here's why you should be happy that Microsoft is embracing Node.js. Luettu: https://www.theguardian.com/technology/blog/2011/nov/09/programming-microsoft. Luettu 11.2.2021.

Digitransit 2021. High-frequency positioning. Luettavissa: https://digitransit.fi/en/developers/apis/4-realtime-api/vehicle-positions/. Luettu 1.3.2021.

Flanagan, David 2020. JavaScript: The Definitive Guide, Seventh Edition. O'Reilly Media Inc. USA.

HiveMQ Team 2015. Introducing the MQTT Protocol – MQTT Essentials: Part 1. Luettavissa: https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/. Luettu: 15.2.2021.

Lynkova, Darina 2019. IoT Statistics and Trends to Know in 2020. Leftronic. Luettavissa: https://leftronic.com/internet-of-things-statistics/. Luettu 3.2.2021.

Morrison, J Paul s.a. Flow-based Programming. Luettavissa: https://jpaulm.github.io/fbp/index.html. Luettu 3.2.2021.

Npm 2021. About npm. Luettavissa: https://docs.npmjs.com/about-npm. Luettu 11.2.2021.

OASIS Open 2015. MQTT Version 3.1.1 Plus Errata 01. Luettavissa: https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/errata01/os/mqtt-v3.1.1-errata01-os-complete.html#_Toc442180821. Luettu 12.2.2021.

OpenJS Foundation 2021a. Node-RED: About. Luettavissa: https://nodered.org/about/. Luettu: 1.2.2021.

OpenJS Foundation 2021b. About Node.js. Luettavissa: https://nodejs.org/en/about/. Luettu 11.2.2021.

Raspberry PI Foundation 2021. Raspberry Pi Documentation. Luettavissa: https://www.raspberrypi.org/documentation/. Luettu 1.2.2021.

Raspberry Pi Foundation 2021a. Raspberry Pi 4 Tech Specs. Luettavissa: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/. Luettu 4.2.2021.

Raspberry Pi Foundation 2021b. Updating and upgrading Raspberry Pi OS. Luettavissa: https://www.raspberrypi.org/documentation/raspbian/updating.md. Luettu 18.2.2021.

Raspberry Pi Foundation 2021c. Securing your Rapsberry Pi. Luettavissa: https://www.raspberrypi.org/documentation/configuration/security.md. Luettu 20.2.2021.

Raspberry Pi Trading Ltd. 2019. Raspberry Pi 4 Computer Model B: Product Brief. Luettavissa: https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-4-Product-Brief.pdf. Luettu 4.2.2021.

Scott, Bill 2013. Clash of the Titans: Releasing the Kraken | NodeJS @paypal. Katsottavissa: https://www.youtube.com/watch?v=tZWGb0HU2QM. Katsottu 11.2.2021.

Spring, Tom 2017. Attackers Use Typo-Squatting To Steal npm Credentials. Luettavissa: https://threatpost.com/attackers-use-typo-squatting-to-steal-npm-credentials/127235/. Luettu 11.2.2021.

Thesen, Arne 1978. Computer Methods in Operations Research. Academic Press Inc. Lontoo.

Veracode 2021. JavaScript Security: What is JavaScript. Luettavissa: https://www.veracode.com/security/javascript-security. Luettu 11.2.2021.

Xiao, Yunong 2014. Node.js in Flames. Luettavissa: https://netflixtechblog.com/node-js-in-flames-ddd073803aa4. Luettu 11.2.2021.