



VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Matti Rintamäki

IOT-LAITTEEN KÄYTTÖLIITTYMÄT JA PILVIPALVELUUN LIITTÄMINEN TIETOTURVA HUOMIOIDEN

Tekniikka
2021

TIIVISTELMÄ

Tekijä	Matti Rintamäki
Opinnäytetyön nimi	IoT-laitteen käyttöliittymät ja pilvipalveluun liittäminen tietoturva huomioiden
Vuosi	2021
Kieli	suomi
Sivumäärä	49
Ohjaaja	Timo Kankaanpää

Opinnäytetyön aiheena oli etsiä TJK Tietolaite Oy:lle ratkaisuvaihtoehtoja elektronikkalaitteen liittämiseksi pilvipalveluun. Laitteen paikallista käyttöliittymää vastaava etäkäyttöliittymä tuli olla helposti tuotavissa esimerkiksi PC:lle tai älypuhelimelle. Molemmista käyttöliittymistä tuli voida suorittaa samat ohjaus- ja valvontatoiminnot.

Isossa osassa opinnäytetyötä oli myös käyttää jotain julkista pilvipalvelua, jossa toimivat viestinvälitysohjelmisto, etäkäyttöliittymä, tietokanta ja kaikki edellä mainittujen vaatimat ohjelmistot.

Keskeisenä vaatimuksena oli myös, että järjestelmän tietoturvapuoli on otettu huomioon. Tässä projektissa tietoturva sisälsi kaiken tarpeellisen suojaamisen salastuksen sekä salauksen käyttämisen viestinnässä.

Opinnäytetyön tuloksena oli yritykselle dokumentaatio IoT-viestintäprotokollalla toimivan verkon pystyttämiseen ja perusteet sellaisen verkon eri osien valitsemiseen ja niiden säätämiseen sekä prototyyppi tämänlaisesta järjestelmästä.

ABSTRACT

Author	Matti Rintamäki
Title	Interfaces of IoT Device and a Secure Connection to a Cloud Service.
Year	2021
Language	Finnish
Pages	49
Name of Supervisor	Timo Kankaanpää

The subject of this thesis was to look for the best solutions for TJK Tietolaite Oy to connect an electronic devices to a cloud service. Device's remote user interface had to have the same set of functions as the local interface, and it had to be easily accessed from a PC or smartphone.

A major requirement of the thesis was to use some public cloud service. The cloud service would be running the required software for a message broker, remote interface and database.

Other major requirement of the thesis was to make sure that the network was safe. The safety in this project included protecting all necessary parts of the system with passwords and the use of encryption in communications.

The result of the thesis is documentation for the company to set up a network operating with the internet of things communication protocol and foundation for selecting and configuring different parts of such a network. Another result is a prototype of such a system.

Keywords	Communications, cloud service, user interface, data security and embedded system
----------	--

SISÄLLYS

TIIVISTELMÄ

ABSTRACT

KUVA- JA TAULUKKOLUETTELO

KÄSITELUETTELO

1	JOHDANTO.....	8
2	IOT-VERKON YLEISKUVA.....	9
2.1	Käyttökohteet.....	10
2.2	IoT-laitteisto.....	11
2.3	Ohjelmistot.....	12
2.3.1	Käyttöjärjestelmä	12
2.3.2	Viestintäprotokolla.....	12
3	KOMMUNIKOINTI	13
3.1	Viestintäprotokolla.....	13
3.1.1	MQTT	15
3.1.2	DDS.....	17
3.1.3	AMQP	18
3.1.4	XMPP.....	19
3.1.5	Protokollien vertailu.....	19
3.2	VPN.....	21
4	PILVIPALVELU	23
4.1	Palveluntarjoajat	23
4.1.1	Amazon	23
4.1.2	Microsoft.....	23
4.1.3	Google	24
4.1.4	Yhteenveto pilvipalvelujen tarjoajista.....	24
4.2	Tietokanta	25
4.2.1	SQLite	25
4.2.2	MySQL.....	25
4.2.3	PostgreSQL	26

4.2.4	Tietokantojen yhteenveto	26
4.3	Palvelimen ohjelmistot.....	26
4.3.1	Viestinvälittäjäohjelmisto	26
4.3.2	Protokollan ja tietokannan välinen ohjelmisto.....	27
4.3.3	Päivitysohjelmisto	28
4.3.4	Valvontaohjelmisto	28
5	KÄYTTÖLIITTYMÄ	29
5.1	Suunnitelma	29
5.1.1	Toiminnot.....	29
5.1.2	Laajennettavuus.....	29
5.1.3	Ulkoasu	29
5.1.4	Suunnitelma	30
5.1.5	Kirjautuminen	32
6	TIETOTURVA.....	33
6.1	Päivitykset.....	33
6.2	Salaus	33
6.3	Virheentarkistus	33
6.4	Käyttäjän todennus.....	34
7	PROTOTYYPPI	35
7.1	Vaatimusmäärittely	35
7.2	Laitteisto	35
7.3	Ohjelmisto.....	36
7.4	Asennus.....	36
7.4.1	Verkko.....	37
7.4.2	Paikallinen käyttöliittymä	38
7.4.3	Amazon Web Services	40
7.4.4	Etäkäyttöliittymä	41
7.4.5	Tietokanta.....	42
7.5	Testaus	43
7.5.1	Testisuunnitelma	43
7.5.2	Yksikkötestaus	44
7.5.3	Integraatiotestaus.....	45

7.5.4	Järjestelmätestaus	45
8	JOHTOPÄÄTÖKSET	47
	LÄHTEET	48

KUVA- JA TAULUKKOLUETTELO

Kuva 1. Yksinkertainen IoT-laitteen sisältävä verkko	9
Kuva 2. Realistisempi verkko, jossa on laitteita ilman kiinteää IP-osoitetta	10
Kuva 3. Kodin hallinta- ja valvontajärjestelmä	11
Kuva 4. Julkaisuihin ja tilauksiin perustuva viestintämalli	14
Kuva 5. Pyyntöihin ja vastauksiin perustuva viestintämalli	14
Kuva 6. Virtuaalisen erillisverkon toimintaperiaate	22
Kuva 7. MQTT protokollan etu- ja taustaohjelmisto sekä viestin säilytys.	27
Kuva 8. Työpöytäkäyttöliittymän suunnitelma.	30
Kuva 9. Mobiilikäyttöliittymän suunnitelma.	31
Kuva 10. Mosquitto Broker ohjelmiston näkymä	37
Kuva 11. Prototyypijärjestelmän arkkitehtuuri	38
Kuva 12. Paikallinen käyttöliittymä ennen välittäjään yhdistämistä	39
Kuva 13. Paikallinen käyttöliittymä kirjautumisen jälkeen	40
Kuva 14. Etäkäyttöliittymä ilman yhteyttä välittäjään	41
Kuva 15. Etäkäyttöliittymä yhdistettynä välitysohjelmistoon	42
Kuva 16. Tietokannan MQTT-asiakkaan arkkitehtuuri	43
Taulukko 1. Kolmitasoinen palvelunlaadun takaus	16
Taulukko 2. DDS:n palvelunlaaduntakaus käytännöt	17
Taulukko 3. Protokollien ominaisuudet	20
Taulukko 4. Järjestelmän vaatimukset	35

KÄSITELUETTELO

IoT	Internet of things, erilaisia internettiin kytkettyjä laitteita jotka suorittavat automaattista tiedonsiirtoa sekä joiden valvonta ja ohjaus voidaan suorittaa verkon kautta.
IP-osoite	Internet Protocol, numerosarjan muodossa esitetty osoite jolla yksilöidään IP-verkkoihin kytketyt verkkosovittimet.
VPN	Virtual Private Network, Virtuaalinen erillisverkko, tapa yhdistää yksityisiä verkkoja julkisen verkon ylitse luoden niistä virtuaalisen yksityisen verkon.
NAT	Network Address Translation, osoitteenmuunnos, jonka ansiosta yhden julkisen IP-osoitteen takana voi olla useampia laitteita.
ISP	Internet Service Provider, Internet-palveluntarjoaja.
Linux jakelu-versio	Linux Distribution, Linux kerneliin pohjautuva, käyttöjärjestelmän koostava sovelluspaketti.
RTOS	Real-Time Operating System, Reaaliaikainen käyttöjärjestelmä, käyttöjärjestelmä, joka noudattaa reaaliaikaisuuden vaatimuksia.
MQTT	Message Queue Telemetry Transport, OASIS- ja ISO/IEC20922-standardien mukainen IoT-viestintäprotokolla.
DDS	Data Distribution Service, avoimen lähdekoodin, C++-kieleen perustuva, laitteelta laitteelle viestintäprotokolla.
AMQP	Advanced Message Queuing Protocol, eräs IoT-viestintäprotokolla.
XMPP	Extensive Messaging and Presence Protocol, XML:ään perustuva viestintäprotokolla.
XML	Extensible Markup Language, merkintäkielien standardi.
P2P	Peer-to-Peer, vertaisverkko, verkko, jossa ei ole palvelimia ja asiakkaita, vaan jokainen verkon jäsenet siirtävät dataa keskenään
Välityspalvelin	Proxy Server, Välityspalvelin, välittää asiakkaan pyytämän resurssin ja peittää asiakkaan alkuperän.

TLS	Transport Layer Security, salausprotokolla IP-verkon yli kulkevan tietoliikenteen suojaamiseksi.
DTLS	Datagram Transport Layer Security, TLS:ään ja datagrammeihin perustuva, UDP:n päällä toimiva kommunikaatioprotokolla.
DDS Security	DDS-viestintäprotokollan oma salaus toteutus
SASL	Simple Authentication and Security Layer, todennusta ja datan turvaamista varten kehitetty ohjelmistokehys.
Broker	Viestinnässä käytetty keskeinen datalähde, joka vastaanottaa ja välittää viestejä vastaanottajille.
TCP	Transmission Control Protocol, tietoliikenneprotokolla, joka toimittaa luotettavasti datavirran oktetteina ja tarkastaa toimitetun datan virheiden varalta.
UDP	User Datagram Protocol, tietoliikenneprotokolla, joka perustuu yhteydettömään kommunikointiin mahdollisimman pienillä protokollan mekanismeilla.
ODBC	Open Database Connectivity, tietokantajärjestelmien ohjelmointirajapinta.
ACID	Atomicity, Consistency, Isolation and Durability, sarja tietokannan transaktio ominaisuuksia joiden tarkoitus on taata datan oikeellisuuden virheistä huolimatta.
HTML	Hypertext Markup Language, yleisesti internetsivustojen luomiseen käytetty kuvauskieli.
CSS	Cascading Style Sheets, World Wide Web Consortiumin ylläpitämä WWW-dokumenttien tyyliohjeiden laji.
T-Plat.E	Tietolaite Platform Embedded, Tietolaite Oy:n oma sulautettujen järjestelmien ohjelmistokehitysalusta.
WebSocket	Viestiprotokolla, joka sallii kaksisuuntaisen kommunikaation yhden TCP-yhteyden yli.
lwIP	Lightweight Internet Protocol, avoimen lähdekoodin TCP/IP-pino, joka on erityisesti suunniteltu sulautetuille järjestelmille.

1 JOHDANTO

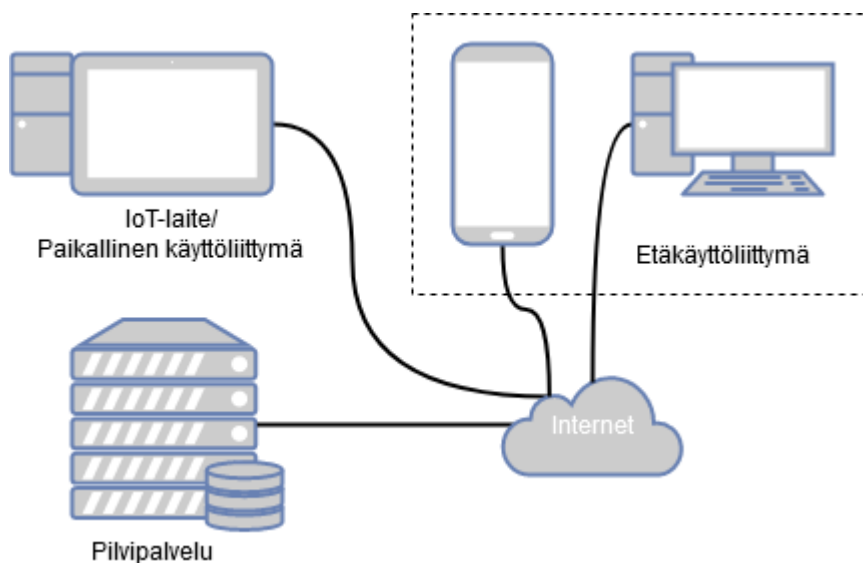
Tämän opinnäytetyö tarkoitus on nostaa TJK Tietolaite Oy:n valmiutta vastata alati kasvavaan kysyntään IoT-tuotteista. Tavoitteena on löytää parhaat ratkaisukeinot toteuttaa IoT-laitteen paikallinen- ja etäkäyttöliittymä sekä laitteen ja pilvipalvelun välinen viestintä ja toteuttaa järjestelmästä prototyyppi. Työssä tutkitaan kaikki tämänlaisen järjestelmän eri osa-alueet, kuten protokollat, laitteistot ja sovellukset.

Lopputuloksena tulee olla järjestelmä, jossa samat ohjaus- ja valvontatoiminnot voidaan suorittaa paikallisesta ja etäkäyttöliittymästä mistä tahansa verkkoselaimen sisältävästä, internettiin kytketystä laitteesta. Järjestelmän täytyy myös tallettaa tiedot tietokantaan ja olla tietoturvallinen, joten järjestelmään täytyy suunnitella päivitysominaisuus, mikä estää vanhentuneen ohjelmiston käytön.

Työ koostuu tutkimus-, asennus- ja testausvaiheista. Tutkimusvaiheessa selvitetään järjestelmän eri osien vaatimukset ja vaihtoehdot sekä perehdytään syvällisemmin eri vaihtoehtoihin ja niiden ominaisuuksiin. Tutkimusvaiheessa myös suunnitellaan järjestelmän käyttöliittymä. Asennusvaiheessa tutkimusvaiheessa valitut ohjelmistot asennetaan ja säädetään. Testausvaiheessa järjestelmää koeajetaan kovalla rasituksella. Koeajossa järjestelmän käyttäytymistä tarkkaillaan ja säädetään tarpeen mukaan.

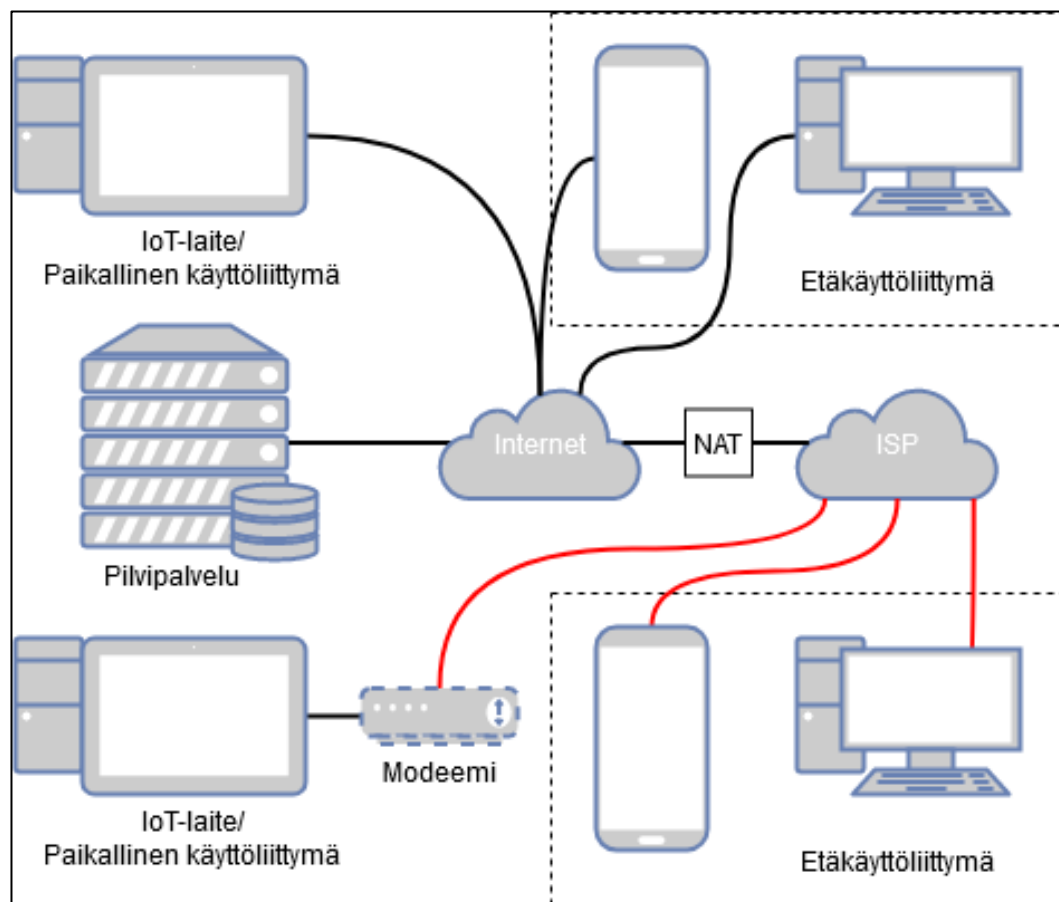
2 IOT-VERKON YLEISKUVA

Tässä luvussa avataan IoT-laitteita sisältävän verkon toimintaperiaatetta, esimerkiksi käyttökohteista, haasteita ja huomioon otettavia asioita. Yksinkertaisimmillaan internetin yli toimiva IoT-verkko on kuvan 1 tapainen ja koostuu valvonta/ohjaus IoT-laitteesta, joka on kytketty internettiin, pilvipalvelusta, jonka tietokantaa data arkistoidaan ja laitteesta, jolta etäkäyttöliittymää käytetään.



Kuva 1. Yksinkertainen IoT-laitteen sisältävä verkko

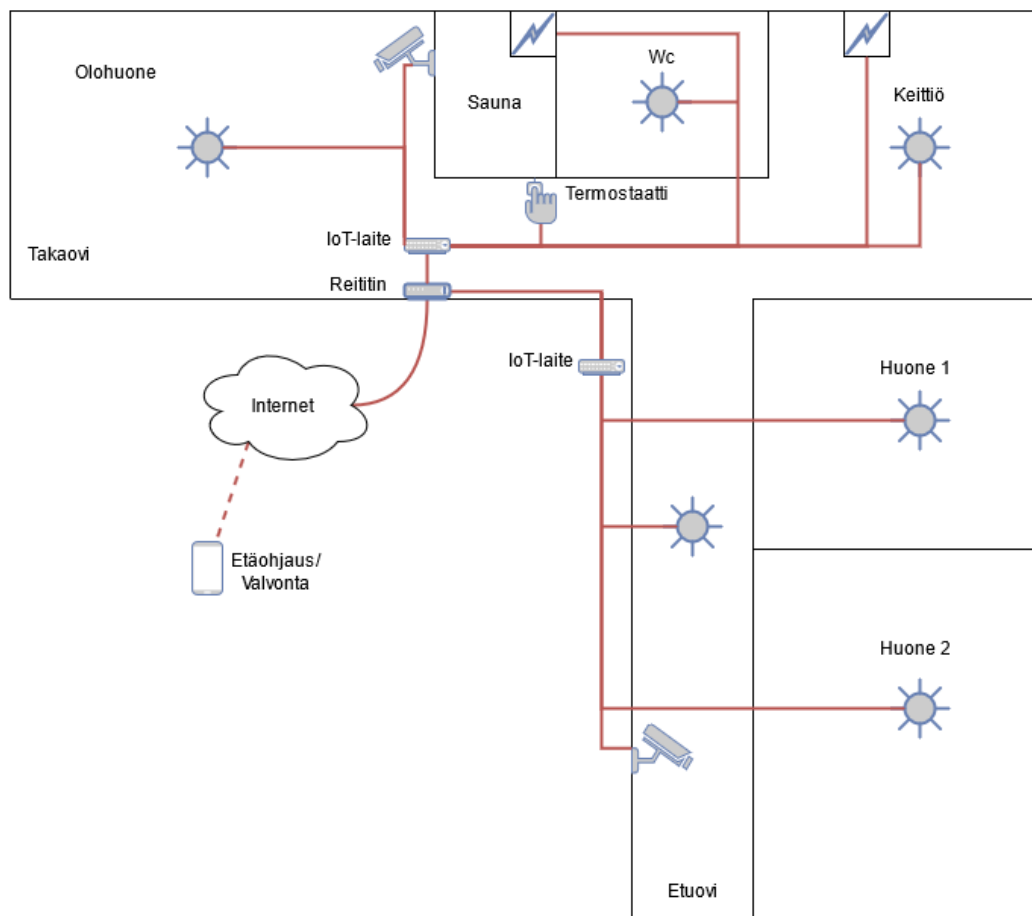
Kuvan 1 kaltainen verkko on kuitenkin epätodennäköinen, sillä se vaatisi, että kaikilla verkkoon kytketyistä laitteista on kiinteät IP-osoitteet. Todennäköisemmin verkko saattaisi olla kuvan 2 tyylinen, jossa on useampi IoT-laite ja osalla verkkoon kytketyistä laitteista on kiinteät IP-osoitteet, toisilla ei. Tällöin toteutus vaatii sen, että käytössä on esimerkiksi VPN, joka luo virtuaalisen lähiverkon päätelaitteiden ja pilvipalvelun välille.



Kuva 2. Realistisempi verkko, jossa on laitteita ilman kiinteää IP-osoitetta

2.1 Käyttökohteet

Käyttökohteita IoT-verkoilla on useita, esimerkiksi valvonta- ja hallintajärjestelmä, joka sisältää kameroita, liiketunnistimia, termostaatteja ja erillisten sähkölaitteiden ohjaamista. Muita käyttötarkoituksia ovat esimerkiksi älykasvihuone, jossa kastelu, valaistus ja lämpötila ovat etäohjattavia, tai teollisuusautomaatio, jolla halutaan valvoa esimerkiksi tuotevirtaa, inventaariota, laatua ja turvallisuutta. Kuvassa 3 on havainnollistettuna kotiin asennettu hallinta- ja valvontajärjestelmä.



Kuva 3. Kodin hallinta- ja valvontajärjestelmä

2.2 IoT-laitteisto

Laitteisto perustuu useimmiten käyttötarkoitusta varten kehitettyyn elektroniikkaan, sillä käyttötarkoitukset ovat usein hyvin erilaisia. Toisinaan laitteisto saattaa kuitenkin perustua esimerkiksi sulautettujen järjestelmien kehitysalustaan tai teollisuus PC:seen. Laitteisto on vastuussa tietojen keräämisestä, tietojen toimittamisesta sekä säätökäskyjen vastaanottamisesta ja suorittamisesta. Kerätyt tiedot voivat olla esimerkiksi lämpötila-anturista, valoanturista, virta-anturista tai kuvadataa kamerasta. Ohjauskäskyt voivat olla vaikka valojen päälle ja pois päältä kytkemistä, termostaatin säätämistä tai sähkölaitteiden sähkönsyötön katkaiseminen. /1/

2.3 Ohjelmistot

Tässä kappaleessa käydään läpi IoT-laitteen ohjelmistovaatimuksia. IoT-laite vaatii käyttötarkoituksen mukaisen ohjelmiston. Käyttötarkoituksia on paljon, aina älyjääkaapeista tehdasautomaatiikkaan, joten ei voi olla yhtä joka tapaukseen sopivaa ohjelmistopakettia. Saatavilla on kuitenkin käyttötarkoituksen perusteella valittavia valmiita paketteja, jotka sisältävät kyseiseen käyttötarkoitukseen kaiken tarpeellisen, jolloin käyttöönotto on erittäin vaivatonta.

2.3.1 Käyttöjärjestelmä

Käyttöjärjestelmä IoT-laitteissa on useissa tapauksissa hyödyllinen, joskaan ei pakollinen. Esimerkiksi laite, joka ottaa mittatietoja ylös ja lähettää ne käyttäen sarjayhteyttä tai ethernet-yhteyttä, on toteutettavissa myös täysin ilman käyttöjärjestelmää.

Käyttöjärjestelmänä IoT-laitteissa suositetaan reaaliaikakäyttöjärjestelmiä (RTOS), mutta nykypäivänä myös Linuxin eri jakeluversioita, esimerkiksi Ubuntu Core ja Embedded Linux. Käyttöjärjestelmä tulisi valita sen perusteella, mitä laitteen halutaan tekevän ja mille käyttöjärjestelmälle tarvittavat sovellukset löytyvät. Kehittämisen kannalta Linux jakeluversiot ovat varmasti helpoin valinta, mutta ne vaativat prosessorilta ja tallennustilalta paljon enemmän kuin reaaliaikakäyttöjärjestelmät.

/2/

2.3.2 Viestintäprotokolla

Käyttöjärjestelmän lisäksi tarvitaan muutakin ohjelmistoa. Viestintäprotokolla on vastuussa laitteen ja palvelimen välisestä liikennöinnistä ja viestien toimitusvarmuudesta. Tämänlaisia protokollia ovat esimerkiksi MQTT, DDS, AMQP ja XMPP.

3 KOMMUNIKOINTI

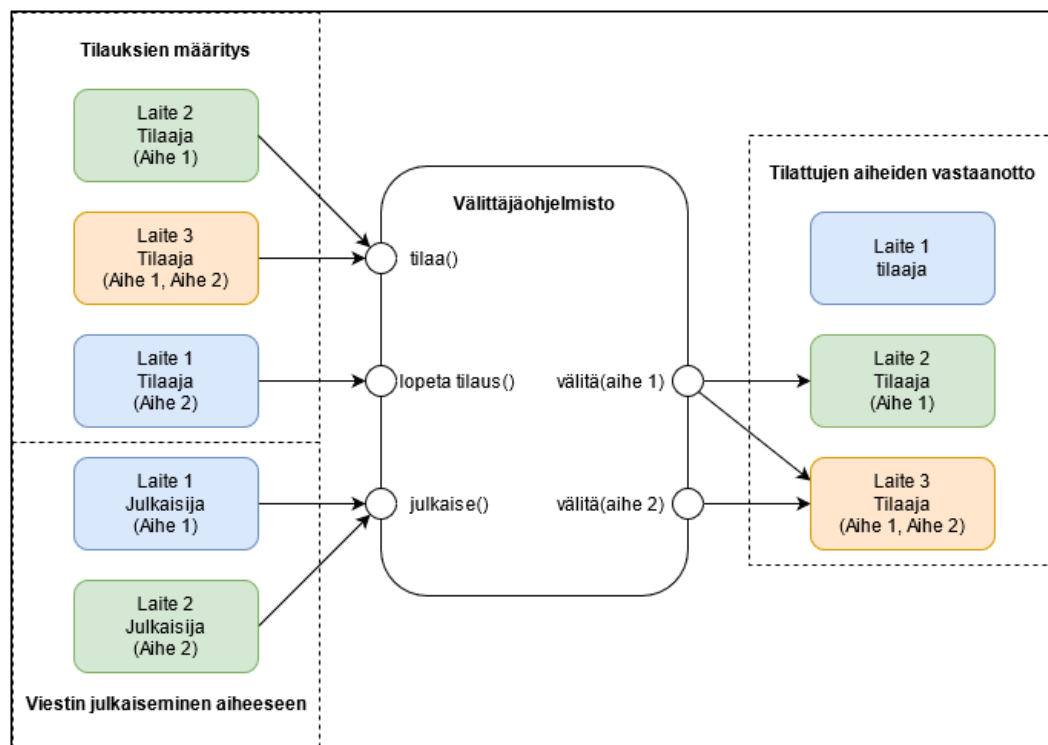
Tässä luvussa käydään läpi laitteiden välistä kommunikaatiota verkon yli ja sen toteuttamista. Toteutus vaatii ainakin jonkin protokollan, jotta kaikki laitteet tietävät säännöt, joiden puitteissa tietoa siirretään. Laitteet saattavat myös tarvita VPN-yhteyden välilleen, jotta laitteet, joilla ei ole kiinteää IP-osoitetta, kykenevät kommunikoimaan palvelimen kanssa. Tämä on tarpeen jos käytetään viestintäprotokollaa, joka toimii UDP-tietoliikenneprotokollalla.

3.1 Viestintäprotokolla

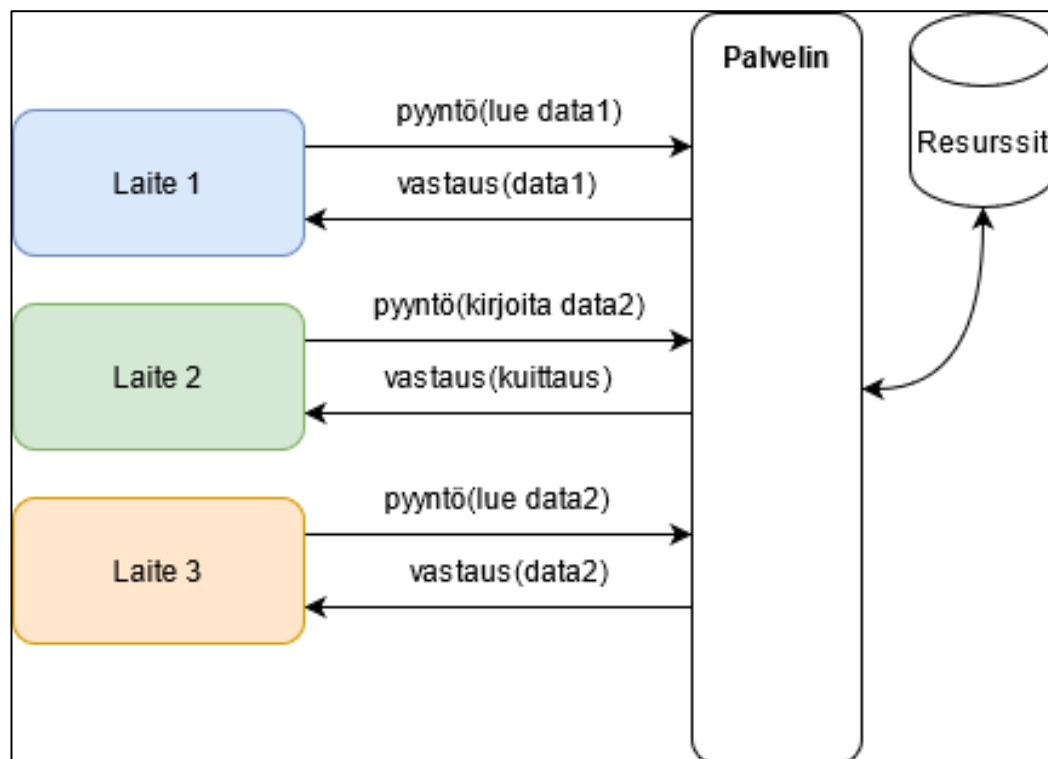
Viestintäprotokollan vastuulla on laitteiden välinen kommunikointi. Tämän työn tapauksessa, viestintäprotokolla hoitaa IoT-laitteen ja pilvipalvelun, pilvipalvelun ja etäkäyttöliittymän sekä pilvipalvelun ja tietokannan väliset kommunikaatiot.

Protokollan viestintämalli voi perustua moneen eri malliin, mutta yleisimpien protokollien viestintämalli perustuu julkaisuihin ja tilauksiin tai pyyntöihin ja vastauksiin. Julkaisuihin ja tilauksiin perustuvassa viestintämallissa julkaisija kirjoittaa viestejä aiheeseen ja tilaaja tilaa aiheen. Viestinvälitysohjelmisto toimittaa tilattuun aiheeseen julkaistut viestit tilaajille. Tässä mallissa tilaajan ei siis tarvitse erikseen pyytää tietoa, vaan tiedot toimitetaan tilaajalle aina uuden julkaisun jälkeen spontaanisti. Sama asiakas voi julkaista ja tilata monia aiheita. Kuvassa 4 on malli julkaisuihin ja tilauksiin perustuvasta viestintämallista.

Pyyntöihin ja vastauksiin perustuva viestintämalli on yksinkertaisempi, mutta asioiden tekeminen vaatii enemmän kommunikointia, kuin julkaisuihin ja tilauksiin perustuvassa järjestelmässä, sillä aina jos halutaan lukea tietoa, laitteen pitää sitä erikseen pyytää. Tässä mallissa, palvelin hoitaa datan kirjaamisen omiin resursseihin ja lukemisen omista resursseistaan. Kuvassa 5 on kaavio pyyntöihin ja vastauksiin perustuvan viestintämallin toimintaperiaatteesta. /3/



Kuva 4. Julkaisuihin ja tilauksiin perustuva viestintämalli



Kuva 5. Pyyntöihin ja vastauksiin perustuva viestintämalli

Tämän opinnäytetyön tapauksessa, tilauksiin ja julkaisuihin perustuva protokolla on käytännöllisempi, sillä dataa aiotaan lähettää isoja määriä joten datan pyytämisen aiheuttaisi paljon turhaa verkkoliikennettä.

IoT-käyttöön tarkoitettuja viestintäprotokollia on useita, joista tämän opinnäytetyön vertailussa ovat MQTT, DDS, AMQP ja XMPP. Protokollan valinnassa tärkeässä osassa ovat helppokäyttöisyys, resurssien käyttö, tuki useille alustoille skaalautuvuuden takia sekä toimintavarmuus viestien perille saamiseksi. Tässä kappaleessa tutkitaan eri protokollien ominaisuuksia ja vertaillaan niitä keskenään.

3.1.1 MQTT

MQTT on kevyt ja suorituskykyinen julkaisuihin ja tilaamiseen perustuva viestintäprotokolla, jonka resurssien käyttö on erittäin pieni, mikä tekee siitä erittäin käytökelpoisen pienillä mikrokontrollereille. MQTT tukee myös viestienvälittäjäohjelmistoja. MQTT:ssä on sisäänrakennettu palvelunlaadun takaus, jossa on kolme eri tasoa. Taulukossa 2, MQTT:n palvelunlaadun takauksen tasot ja niiden vertailua.

/4/

Taulukko 1. Kolmitasoinen palvelunlaadun takaus

Taso	Selite
0 – Vain kerran	Tämä taso takaa, että vastaanottaja saa viestin enintään yhden kerran. Viesti lähetetään, mutta viestin perille toimitusta ei varmenneta
	Hyvät puolet
	Taso 0 on erittäin kevyt verkon käytön suhteen. Viesti lähetetään vain kerran joten duplikaatit ovat mahdottomuus.
	Huonot puolet
	Taso 0 on erittäin epävarma toimitusmenetelmä. Jos yhteys katkeaa, paketit eivät mene perille ja siitä ei saada mitään tietoa.
1 – Vähintään kerran	Selite
	Tämä taso takaa että kaikki vastaanottajat saavat viestin vähintään yhden kerran. Viestin vastaanotettuaan vastaanottajat vastaavat lähettämällä PUBACK-paketin lähettäjälle. Lähettäjä lähettää viestiä kaikille vastaanottajille niin kauan, että kaikki vastaanottajat ovat vastanneet viestiin.
	Hyvät puolet
	Taso 1 takaa, että viestiä lähetetään kunnes viesti tiensä perille vähintään kerran. Tämä taso on myös aika kevyt verkon käytön suhteen
	Huonot puolet
	Tason 1 huono puoli on, että jos joku vastaanottajista ei saa viestiä tai vastaanottajan PUBACK- paketti ei löydä jostain syystä tietään perille, viesti julkaistaan niin monta kertaa, että kaikki vastaanottajat vastaavat
2 – Enintään kerran	Selite
	Tämä taso takaa, että kaikki vastaanottajat saavat paketin vain yhden kerran. Viestin välityksen yhteydessä laitteet suorittavat neljä osaisen kättelyn, lähettäjä lähettää paketin, vastaanottaja vastaa PUBREC-paketilla saaneensa paketin, lähettäjä vahvistaa PUBREL-paketilla saaneensa tämän tiedon johon vastaanottaja vastaa vielä PUBCOMP-paketilla vahvistaakseen
	Hyvät puolet
	Taso 2 takaa, että kaikki vastaanottajat saavat viestin, mutta yksikään vastaanottajista ei saa duplikaatteja
	Huonot puolet
	Tason 2 huonona puolena on raskas verkon käyttö, jokaista viestiä varten lähetetään kaksi viestiä molempiin suuntiin

3.1.2 DDS

DDS on julkaisuihin ja tilaamiseen perustuva viestintäprotokolla. DDS koostuu viidestä peruskomponentista: aihe (Topic), julkaisija (Publisher), tilaaja (Subscriber), datan kirjoittaja (DataWriter) ja datan lukija (DataReader). Aihe pitää sisällään tiedon yhdestä datatyypistä, julkaisuista ja näytteiden saatavuudesta. Julkaisija ohjaa ja rajoittaa datan kirjoittajan toimittamaa tietoa ja tilaaja ohjaa ja rajoittaa datan lukijan vastaanottamaa tietoa. Yhdellä asiakkaalla voi olla monia datan kirjoittajia ja monia datan lukijoita, mutta datan kirjoittajalla ja lukijalla voi olla vain yksi aihe. DDS:n palvelunlaadun takaus perustuu sarjaan käytäntöjä, joista voi valita tarvitsemansa ominaisuudet, käytännöt ovat selitettynä taulukossa 1. Muista protokollista poiketen DDS on datakeskeinen. /5/

Taulukko 2. DDS:n palvelunlaaduntakaus käytännöt

Käytäntö	Selite
DEADLINE	Julkaisijaa vaaditaan lähettämään vähintään yksi arvo aikarajan sisällä.
RELIABILITY	Käyttövarmuus asetus saattaa uudelleen lähettää tai estää datan lähetystä tarpeen mukaan. Estämistä voi ohjata max_blocking_time muuttujalla.
LIFESPAN	Määrittää näytteiden elinkaaren ettei laite vastaanota liian vanhoja näytteitä.
RESOURCE_LIMITS	Rajoittaa DDS:n muistinkäyttöä määrittämällä maksimi viestijonon pituuden silloin kun HISTORY on asetettu KEEP_ALL tilaan.
PRESENTATION	Määrittää kuinka sovelluksen vastaanottama data esitetään datan lukijoille.
TIME_BASED_FILTER	Määrittää, että dataa voi lähettää vain kerran aikamääreen sisässä.
LIVELINESS	Määrää miten DDS tunnistaa onko data kirjoitin vielä elossa.

OWNERSHIP	Aihe instanssien omistus voi olla jaettu tai yksityinen. Yksityinen omistus estää muita data kirjoittimia lähettämästä aiheen instanssiin liittyvää dataa.
OWNERSHIP_STRENGTH	Jos instanssin omistus on yksityinen, tämä käytäntö sijoittaa instanssiin yhteydessä olevat data kirjoittimet tärkeysjärjestykseen siten, että korkeimman prioriteetin kirjoittimen data luetaan ensisijaisesti.
DESTINATION_ORDER	Määrittää minkä data kirjoittajan arvon tilaaja ottaa lopulliseksi arvoksi tilanteessa, jossa moni data kirjoittaja lähettää samaa arvoa.
HISTORY	Määrittää montako arvoa data kirjoittajat ja lukijat säilyttävät paikallisesti.
DURABILITY	Määrää tallennetaanko data kirjoittajan lähettämät arvot siltä varalta, että ilmaantuu uusia data lukijoita.
DURABILITY_SERVICE	Käytetään vain jos DURABILITY on asetettu arvolle PERSISTENT tai TRANSIENT ja käytössä on Persistence Service. Määrittää paljonko dataa data kirjoitin tallettaa.
WRITER_DATA_LIFECYCLE	Määrittää miten data kirjoitin käsittelee instanssien elinkaarta.
READER_DATA_LIFECYCLE	Määrittää miten data lukija käsittelee instanssien elinkaarta.

3.1.3 AMQP

AMQP–viestintäprotokollaa voidaan käyttää julkaisu ja tilaus-viestintämallin lisäksi pisteestä pisteeseen-mallilla. AMQP vahvuuksia on vahva tietoturva, jonoutaminen ja reaaliaikaisuus. AMQP kykenee laittamaan viestejä jonoon, jos vastaanottajaan ei saada yhteyttä silloin kun viesti lähetetään. MQTT:n tapaan AMQP käyttää myös, taulukon 2, mukaista kolmitasoista palvelunlaadun takausta. /6/

3.1.4 XMPP

XMPP on XML:ään pohjautuva, ensisijaisesti pikaviestintään, ääni- ja videopuheluihin tarkoitettu viestintäprotokolla, joka perustuu palvelin-asiakas-arkkitehtuuriin, mikä tarkoittaa, että XMPP vaatii erillisen palvelimen käyttäjien väliin. XMPP:tä voidaan käyttää myös muun kaltaiseen viestintään. XMPP:ssä ei ole minäänlaista sisäänrakennettua palvelunlaadun takausta, vaan käyttäjä joutuu toteuttamaan viestin välityksen varmistamisen itse. /7/

3.1.5 Protokollien vertailu

Protokollista MQTT, AMQP ja XMPP ovat viestikeskeisiä eli muutoksen viestintä tapahtuu käskyllä, esimerkiksi ”Päivitä anturin #2, mitattu lämpötila arvoon 30 °C”. Viestikeskeisessä välityksessä viestinvälitysohjelmisto ei tiedä sensoreista tai niiden arvoista mitään, jolloin sovelluksien täytyy osata vastaanottaa niille osoitetut käskyt ja toimia niiden mukaan. DDS on datakeskeinen protokolla, eli viestintä tapahtuu ilmoittamalla Sensorin tai säätimen ID ja arvo väliohjelmistolle, joka välittää tiedot niitä tarvitseville verkon solmuille eli tilaajille.

Viestintämallina voidaan käyttää kaikilla protokollilla Publish-Subscribe-mallia, joka perustuu keskeiseen datalähteeseen, viestinvälitysohjelmistoon. Viestinvälitysohjelmisto vastaanottaa julkaisuja toimittajilta, varastoi ne ja toimittaa julkaisut tilaajille. DDS-, AMQP- ja XMPP-protokollissa, voidaan käyttää myös Request-Response-mallia, joka perustuu asiakkaan lähettämiin pyyntöihin, tässä mallissa dataa ei lähetetä ollenkaan automaattisesti.

Viestintäprotokollat jättävät datan siirtämisen tietoliikenneprotokollien vastuulle. Vertailussa olevista protokollista MQTT, AMQP ja XMPP käyttävät TCP-tietoliikenneprotokollaa ja DDS käyttää UDP-protokollaa. TCP-protokollan hyviä puolia ovat sen yhteyden ylläpito, virheentarkistus, toimitusvarmistus ja järjestysvarmuus ja huonoja puolia on raskaus sekä UDP:hen verrattuna hitaus. UDP-protokollan hyviä puolia ovat sen keveys ja nopeus ja huonoja puolia on sen yhteydettömyys, toimitusvarmistus sekä uudelleenlähetys ominaisuuksien puute. /8/ /9/

Palvelunlaadun takaus vaihtelee protokollien välillä, XMPP:ssä käyttäjä joutuu itse rakentamaan palvelunlaadun takausmekanismit, MQTT:ssä ja AMQP:ssa on sisäänrakennettuna kolmitasoinen palvelunlaaduntakaus, joista käyttäjä voi valita suotuisimman. DDS:ssä on 15 erilaista käytäntöä, joita asettamalla käyttäjä voi tarvitsemansa ominaisuudet sisältävän palvelunlaadun takaus järjestelmän.

Salausprotokollana kaikki viestintäprotokollat voivat käyttää TLS:ää. DDS voi käyttää myös DTLS:ää ja DDS:n omaa DDS Securityä. XMPP voi käyttää TLS:n lisäksi SASL:ää.

Taulukossa 3 vertaillaan eri protokollien ominaisuuksia.

Taulukko 3. Protokollien ominaisuudet

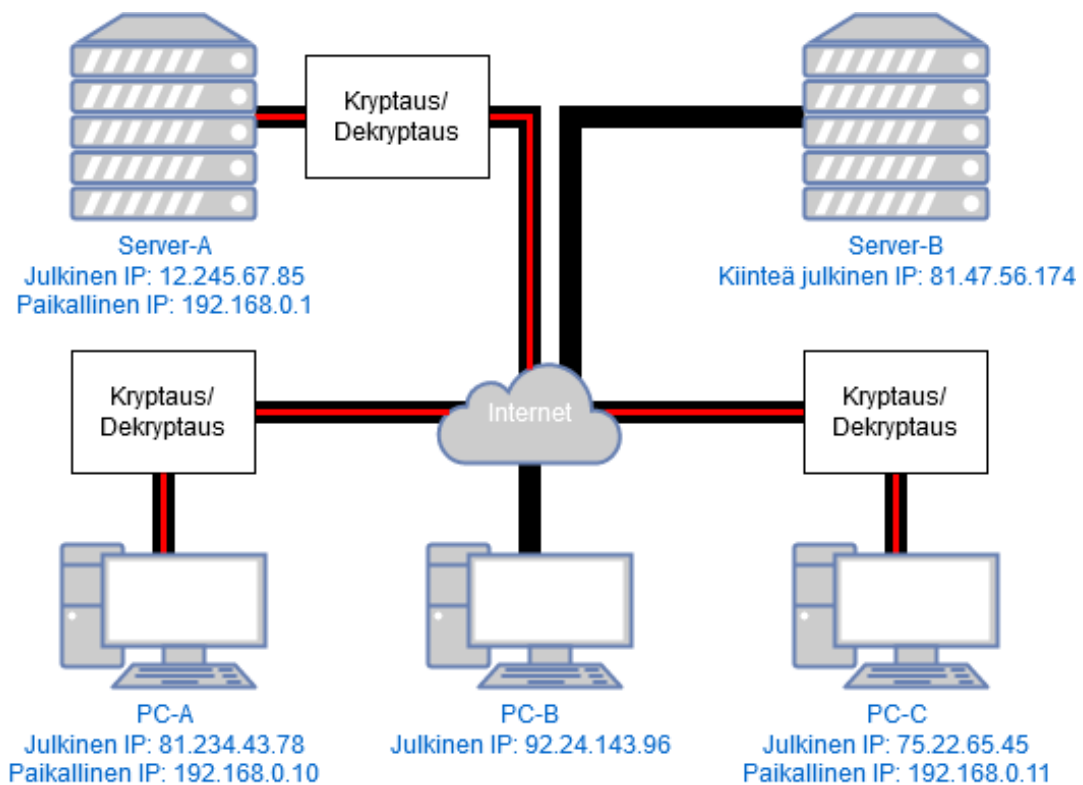
	MQTT	DDS	AMQP	XMPP
Keskeisyys	Viestikeskeinen	Datakeskeinen	Viestikeskeinen	Viestikeskeinen
Viestintämalli	Publish-Subscribe	Publish-Subscribe tai Request-Response	Publish-Subscribe tai Request-Response	Publish-Subscribe tai Request-Response
Tietoliikenneprotokolla	TCP	UDP	TCP	TCP
Palvelunlaatu	kolmitasoinen	15 käytäntöä	kolmitasoinen	Ei sisäänrakennettua
Salausprotokollat	TLS	TLS, DTLS, DDS security	TLS	SASL, TSL

Tässä opinnäytetyössä viestintäprotokollana käytetään MQTT-protokollaa koska se on yksi yleisimmistä ja koska minulla on hieman aiempaa kokemusta sen käyttämisestä.

3.2 VPN

Virtuaalinen erillisverkko on tarpeen tapauksissa, joissa laitteilla ei ole käytössä kiinteitä IP-osoitteita. VPN tunneloi laitteet haluttuun verkkoon, jolloin tiedonsiirrossa voidaan käyttää paikallisia IP-osoitteita. VPN myös luonnostaan nostaa järjestelmän turvallisuutta. VPN luo salatun yhteyden laitteiden välille, käyttäen salausavaimia ja siten tekee yhteyden salakuuntelusta ja sieppauksesta vaikeampaa. Salausavaimien ansiosta, voidaan luoda julkisen verkon yli turvallinen yhteys, jonka yli kulkeva data salataan lähetettäessä kryptaamalla ja vastaanottaessa avataan salausavainta käyttäen. /10/

Kuvassa 6 on havainnollistettuna VPN:än toimintaperiaate, mustalla piirretyt viivat kuvaavat julkisia internetyhteyksiä ja punaiset viivat kuvaavat salattua VPN-verkkoa, joka toimii julkisen internetin yli. VPN-verkkoa käyttävät laitteet voivat viestiä toisilleen käyttäen paikallisia IP-osoitteita, jolloin virtuaalinen verkkokortti kokoaa paketit, kryptaa ne ja lähettää julkista verkkokorttia käyttäen vastaanottajalle. Jos esimerkiksi PC-A hakee tietoa Server-A:lta, PC-A yhdistää Server-A:n yksityiseen IP-osoitteeseen, joka on kuvan tapauksessa 192.168.0.1. Kuvassa PC-B ei ole yhteydessä virtuaaliseen erillisverkkoon, joten se ei näe Server-A:ta, PC-A:ta ja PC-C:tä paikallisina laitteina eikä voi viestiä niille niiden yksityisillä IP-osoitteilla. Server-B:llä taas on kiinteä julkinen IP-osoite joten kaikki laitteet voivat ottaa siihen yhteyden.



Kuva 6. Virtuaalisen erillisverkon toimintaperiaate.

Tässä opinnäytetyössä ei ole tarpeen käyttää virtuaalista erillisverkkoa, sillä pilvipalvelulla on kiinteä IP-osoite ja MQTT toimii auki pidetyn TCP-yhteyden kautta. Yhteys alustetaan tämän takia aina laitteilta palvelimelle.

4 PILVIPALVELU

Tässä luvussa käydään läpi pilvipalvelun viestinnän menetelmiä ja ylläpitoon tarvittavia ohjelmistoja. Pilvipalvelua voidaan käyttää IoT-laitteen ja etäkäyttöliittymän lähettämien viestien tietokantana. Laite julkaisee mittatietoja, joko tasaisin väliajoin tai aina tiedon muuttuessa, riippuen järjestelmän vaatimuksista. Pilvipalvelussa pyörivä sovellus ottaa mittatiedot vastaan ja kirjaa ne ylös tietokantaan. Etäkäyttöliittymät voivat hakea tietokannasta tietoja ja tietokantaan tallennetaan myös kaikki etäkäyttöliittymien lähettämät ohjauskomennot.

4.1 Palveluntarjoajat

Pilvipalveluja on tarjolla monelta eri yritykseltä ja niihin saatavilla olevat palveluita on saatavilla lähes rajattomasti. Tähän kappaleeseen on valittu kolme tämän hetken suosituinta pilvipalvelua.

4.1.1 Amazon

Amazonin tarjoama Amazon Web Services (AWS) on monikäyttöinen pilvipalvelualusta, jonka ominaisuudet soveltuvat esimerkiksi markkinoinnin, tuotannon ja tietoliikenteen tarpeisiin. Projektin tarpeisiin, Amazon tarjoaa palvelut verkkosovelluksen luomiseen ja ylläpitämiseen sekä IoT-tapahtumien seuraamiseen ja datan tietokantaan arkistointiin.

Amazon tarjoaa myös mahdollisuuden käyttää useita palvelunsa ominaisuuksista, joillain rajoituksilla 12 kuukauden ajan, joten kehitysvaiheen aikana ei tarvitse välttämättä maksaa pilvipalvelusta. Amazonin hinnoittelu perustuu valittujen palveluiden hintoihin. Vuonna 2021, Amazonin hintalaskurilla kevyelle käytölle laskettuna, IoT-, tietokanta- ja verkkosovelluspalveluilla hinnaksi tulee 6,56 €/kuukausi. /11/

4.1.2 Microsoft

Microsoftin tarjoama Azure-pilvipalvelu on ominaisuuksiltaan hyvin samankaltainen kuin Amazon Web Services, joskin Azuressa on hieman vähemmän ominaisuuksia. Etuna Azuren valitsemisessa on integraatio muiden Microsoftin tuotteiden

kanssa, jos yritys käyttää jo muita Microsoftin tuotteita, kuten Office 365 ja Teams. Azuresta löytyy ominaisuudet verkkosovelluksiin, IoT-yhteyksiin ja tietokantoihin.

Microsoft tarjoaa Azuren, joillain rajoituksilla, 12 kuukaudeksi ilmaiseen kokeiluun. Microsoftin hinnoitteluperusteet eroavat paljon Amazonin hinnoitteluperusteista, mutta kevyelle käytölle laskettuna, Azure maksaa vuonna 2021 samoilla ominaisuuksilla, Microsoftin omalla laskimella, noin 5,46 €/kuukausi. /12/

4.1.3 Google

Google Cloud Platform on pilvipalvelu, joka toimii samalla infrastruktuurilla kuin Googlen omat järjestelmät, kuten Googlen hakukone, Google Mail ja Google Drive. Myös Googlelta löytyy ominaisuudet verkkosovelluksiin, IoT-yhteyksiin ja tietokantoihin.

Google tarjoaa aloituspakkauksena 300 \$ krediittejä ja 20 täysin ilmaista tuotetta liittymisen yhteydessä. Googlen hintalaskimen mukaan, tarvituilla ominaisuuksilla varustettu, kevyelle käytölle tarkoitettu, pilvipalvelin vuonna 2021 maksaa noin 9,55 €/kuukausi. /13/

4.1.4 Yhteenveto pilvipalvelujen tarjoajista

Amazon Web Services tarjoaa parhaan valikoiman ominaisuuksia esimerkiksi monitorointiin, tietoturvaan ja ominaisuuksien konfigurointiin.

Azuren suurin etu on integraatio muiden Microsoftin työkalujen kanssa ja parhaat työkalut kehitykseen ja testaamiseen.

Googlen etuihin kuuluu laaja avoimen lähdekoodin tuki ja Google Cloud on suunniteltu erityisesti pilvipohjaisille yrityksille.

Projektiin valikoitui ominaisuuksien ja vaivattoman kokeilujakson perusteella Amazonin AWS-pilvipalvelu.

4.2 Tietokanta

Tietokantaa voidaan käyttää esimerkiksi IoT-laitteen ja etäkäyttöliittymän välisenä linkkinä, jolloin tietokantaan tallennetaan laitteen keräämä data ja etäkäyttöliittymän lähettämät ohjauskomennot. Etäkäyttöliittymä hakisi tällaisessa tilanteessa monitoroitavaa datan tietokannasta ja IoT-laite ohjauskomennot.

Koska MQTT on tarkoitettu enimmäkseen viimeisimmän tilatiedon toimittamiseen, on MQTT-välittäjäohjelmistoissa useimmiten ominaisuutena automaattinen viimeisimmän aiheeseen toimitetun viestin lähettäminen, kun jokin laite tilaa aiheen. Tämän vuoksi tässä opinnäytetyössä tietokanta ei toimi linkkinä, vaan lokikirjana. Tietokantaan tallennetaan vastaanotetut MQTT-viestit ja niitä voidaan tarpeen tullen hakea tietokannasta esimerkiksi jos etäkäyttöliittymässä halutaan esittää muutakin dataa kuin viimeisimmät tiedot.

Tietokannan valinnassa on tärkeää ottaa huomioon varastoitavan datan tarpeet, esimerkiksi kuinka paljon dataa siirretään maksimissaan ja jos järjestelmä ohjaa laitteita, vaatiiko ohjaus reaaliaikaisuutta. Valinnassa tärkeää on myös joustavuus dataformaattien kanssa.

4.2.1 SQLite

SQLite on C-kirjastona toteutettu tietokannan hallintajärjestelmä. SQLiten vahvuus on sen pienikokoisuus, optimoituina toteutus vaatii alle 500 kilotavua tallennustilaa. Koska SQLite on C-kirjasto, se linkitetään suoraan sitä käyttävään sovellukseen, mikä tarkoittaa, että ODBC:tä, tietokantapalvelinta tai hallintaohjelmaa ei tarvitse käyttää. Kirjasto on julkinen sovellus joten sen muokkaaminen, vapaa levitys ja ohjelmistoihin linkittäminen on sallittua ilman erillistä lupaa. /14/

4.2.2 MySQL

MySQL on avoimen lähdekoodin relaatiotietokantajärjestelmä jota käytetään paljon web-palveluiden tietokantana. MySQL mahdollistaa tietokantojen luomisen, muokkaamisen ja tietokantojen sisältämän datan talteen ottamisen. Useimmiten MySQL:n rinnalle ohjelmoidaan ohjelma Perl-, PHP- tai Python-ohjelmointikieliä

käyttäen, jonka tehtävänä on automaattisesti syöttää haluttua dataa tietokantaan ja tarvittaessa hakea dataa tietokannasta. /15/

4.2.3 PostgreSQL

PostgreSQL on ilmainen ja avoimen lähdekoodin relaatiotietokantajärjestelmä, joka noudattaa SQL-standardia ja on laajennettavissa monella tapaa. PostgreSQL:n ominaisuuksiin kuuluu data transaktiot ACID-ominaisuuksilla, automaattisesti päivittyvät näkymät, materialisoidut näkymät jotka tallennetaan fyysisenä kopiona määrätyn aikavälein, laukaisimet jotka suorittavat toimintoja, kun laukaisuun vaaditut asiat tapahtuvat, viiteavaimet, joilla voidaan viitata toisessa tietokantataulukossa sijaitsevaan dataan ja tallennetut menetelmät joita muut tietokantaa käyttävät sovellukset voivat kutsua. /16/

4.2.4 Tietokantojen yhteenveto

SQLite ei paikalliseen käyttöön tarkoitetun arkkitehtuurinsa vuoksi sovellu käytettäväksi järjestelmässä, jossa on tarkoitus hakea dataa verkon yli. MySQL ja PostgreSQL ovat ominaisuuksiltaan hyvin lähellä toisiaan, joskin PostgreSQL:n etuina MySQL:ään nähden ovat laajennettavuus, taulukoiden periytyminen, funktioiden ylikuormitus ja standardien mukaisuus. Lukupainotteisessa käytössä MySQL saat-
taa olla PostgreSQL:ää parempi valinta.

Tässä työssä käytetään tietokannan hallintajärjestelmänä avoimen lähdekoodin PostgreSQL tietokantaa, koska yrityksellä on käytössä lisenssi tietokannan kehityssovellukseen ja kokemusta sen käytöstä.

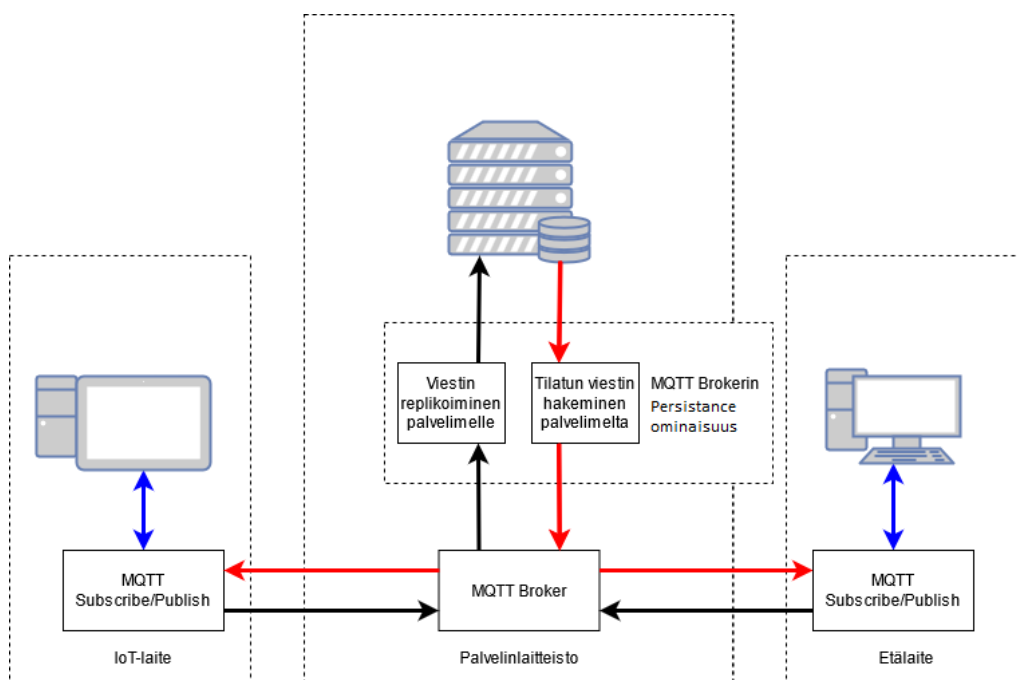
4.3 Palvelimen ohjelmistot

Tässä kappaleessa käydään läpi palvelimelle datan käsittelyyn ja tallentamisen sekä laitteiston ja ohjelmistojen ylläpitoon tarvittavia sovelluksia.

4.3.1 Viestinvälittäjäohjelmisto

Jos verkkoon halutaan kytkeä useita MQTT-asiakkaita, tarvitaan taustaohjelmistoksi jokin viestinvälittäjäohjelmisto. Viestinvälittäjäohjelmiston tehtävä on hoitaa

oikeat tiedot oikeille vastaanottajille. Tällaisia ohjelmistoja ovat esimerkiksi Mosquitto Broker ja EMQ X Broker. Taustaohjelmiston tehtävänä on pitää yllä yhteyksiä tilaajiin ja vastaanottaa julkaisuja asiakkailta sekä toimittaa tilattuihin aiheisiin tulleita julkaisuja niitä tilaaville asiakkaille. Välittäjäohjelmisto myös säilyttää viimeisimmän vastaanotetun viestin valituista aiheista tallessa, jotta uusille tilaajille voidaan toimittaa viimeisin tilatieto. Kuvassa 7, MQTT:n etu- ja taustaohjelmisto sekä viestinsäilytys ominaisuus. /4/



Kuva 7. MQTT protokollan etu- ja taustaohjelmisto sekä viestin säilytys.

Tähän opinnäytetyöhön välittäjäohjelmistoksi valikoitui Mosquitto Broker sen helppokäyttöisyyden ja laajan dokumentaation vuoksi.

4.3.2 Protokollan ja tietokannan välinen ohjelmisto

Palvelimelle tarvitaan sovellus, joka tallentaa julkaistut tiedot tietokantaan, viestintäprotokollaa käyttäen. Joissain tapauksissa sovelluksen voi olla myös kaksisuuntainen, mikä tarkoittaa, että datan tallettamisen lisäksi sovellus kykenee myös hakemaan tietoa tietokannasta. Tämänlaisia sovelluksia on valmiiksi saatavilla protokollasta ja tietokannasta riippuen. Osassa protokollien viestinvälittäjäohjelmistoista

on valmiina ominaisuuksia viestien tietokantaan tallentamista ja tietokannasta hakemista varten.

4.3.3 Päivitysohjelmisto

Palvelimella olisi hyvä olla jokin automatisoitu päivitysohjelmisto, mikä pitäisi palvelimen oman ohjelmiston ajan tasalla.

4.3.4 Valvontaohjelmisto

Laitteiden ylläpitoon olisi hyvä olla sovellus, joka valvoo järjestelmän osien toimintaa, kirjoittaa siitä lokikirjaa ja ilmoittaa ongelmista käyttäjälle, esimerkiksi tekstiviestillä. Yksi tällainen sovellus on Nagios, jonka ominaisuuksiin kuuluu järjestelmän tarkkailu ja ongelmien sekä niihin löytyneiden ratkaisujen ilmoittaminen ylläpitäjälle. Amazon Web Services-pilvipalvelu sisältää myös omat monitorointi ominaisuudet joita voi säätää käyttötarpeen mukaan.

5 KÄYTTÖLIITTYMÄ

Tässä luvussa käydään läpi käyttöliittymän toiminnallisia tarpeita sekä alustavia ulkoasu suunnitelmia. Käyttöliittymässä tulee olla paikallisesti ja etänä samat ohjaus- ja monitorointiominaisuudet. Käyttöliittymän vastuulla on näyttää tietokannasta haettu data käyttäjälle ja toimittaa käyttäjän muuttama data tietokantaan. Tässä opinäytetyössä molemmat käyttöliittymät ovat selainpohjaisia.

5.1 Suunnitelma

Käyttöliittymät toteutetaan HTML-pohjaisena. Etäkäyttöliittymä haetaan laitteelle ottamalla yhteys palvelimen IP-osoitteeseen ja paikallinen käyttöliittymä tulee laitteesta itsestään.

5.1.1 Toiminnot

Käyttöliittymän tulee kyetä ottamaan vastaan monitorointi informaatiota sekä lähettää ohjaukomentoja, jotka voivat olla esimerkiksi päälle/pois komentoja tai jonkun laitteen ohjaukseen liittyvien arvojen muutoksia.

5.1.2 Laajennettavuus

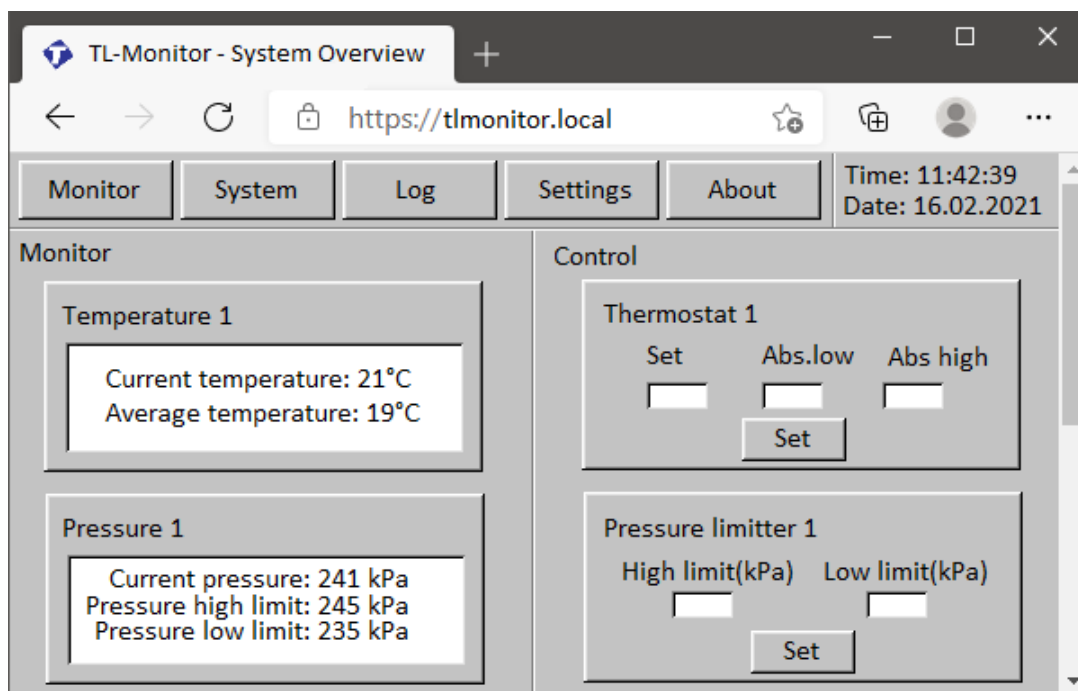
Käyttöliittymään tulee olla helppoa lisätä uusia ohjaus ja monitorointi ominaisuuksia. Laajennettavuuden voi toteuttaa esimerkiksi siten, että muutos tehdään laitteen paikalliseen käyttöliittymään josta se automaattisesti päivittyy palvelimella olevaan etäkäyttöliittymään

5.1.3 Ulkoasu

Ulkoasu tehdään käyttäen CSS:ää, joten ulkoasuja voi olla käyttötarkoituksen mukaan useita eri tyylejä.

5.1.4 Suunnitelma

Koska molemmissa versioissa pitää olla täysin samat ominaisuudet, voidaan molemmissa käyttöliittymissä käyttää samoja HTML- ja CSS-pohjaisia sivustoja. Kuvassa 8 on käyttöliittymän PC-version suunnitelma ja kuvassa 9 mobiilisivuston suunnitelma.



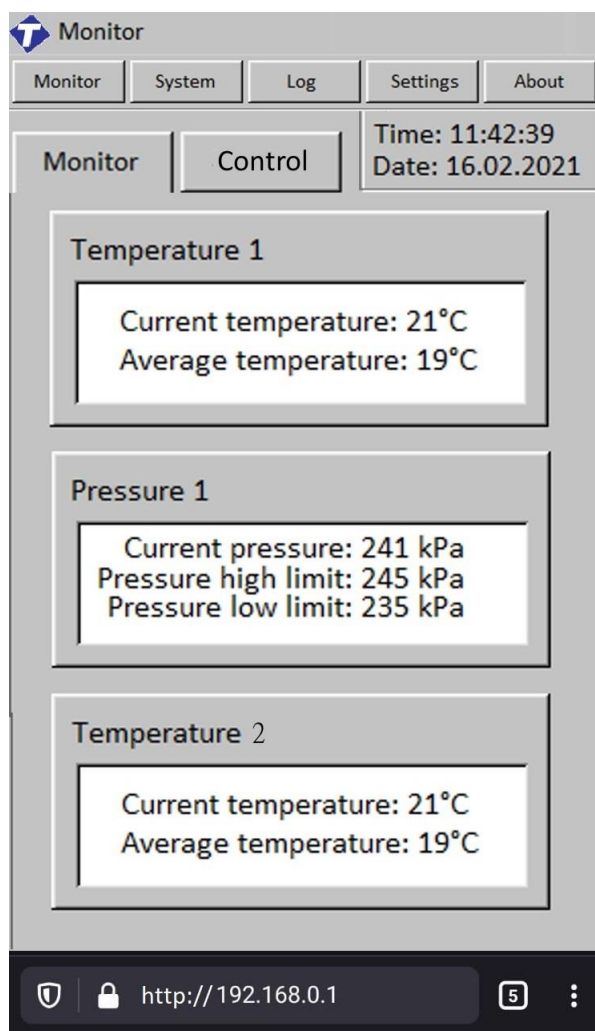
Kuva 8. Työpöytäkäyttöliittymän suunnitelma.

Kuvan 8 suunnitelmassa on karkeasti kuvattuna kaikki käyttöliittymän olennaiset ominaisuudet. Yläpalkista voidaan valita ”Monitor”-linkki, joka avaa valvonta ja ohjaus näkymän. ”System”-linkin takaa saadaan järjestelmän tiedot näkymä. ”Log”-linkin takaa löytyy järjestelmän toimintaan liittyvät lokikirjaukset. ”Settings”-linkin takaa löytyvät käyttöliittymän asetukset kuten ulkoasu asetukset. ”About”-linkin takaa löytyy informaatiota käyttöliittymästä ja pienimuotoinen pika-apu.

Navigaatiopalkin alapuolella on käyttöliittymän toiminnallinen osio, jossa sijaitsevat esimerkiksi valvonta- ja ohjauskortit. Ohjauskorteissa on muokattavia kenttiä ja nappi, joka lähettää kenttiin täytetyn datan ohjattavalle laitteelle.

Jos verkkoselaimen leveys ei ole tarpeeksi suuri, käytetään työpöytäkäyttöliittymässä myös kuvan 9 mukaista välilehti-asetelmaa. Mobiilikäyttöliittymä vastaa työpöytäkäyttöliittymää välilehti-asetelmaa lukuun ottamatta täydellisesti ja niistä pystytään suorittamaan täysin samat toiminnot.

Suunnitelmien ulkonäkö ei ole lopullinen, suunnitelmien tarkoitus on ainoastaan kuvata käyttöliittymien toimintoja.



Kuva 9. Mobiilikäyttöliittymän suunnitelma.

5.1.5 Kirjautuminen

Käyttöliittymässä tulee olla käyttäjän todentamista varten, sisäänkirjautuminen. Sisäänkirjautuminen tapahtuu ennen kuin käyttäjä näkee mitään tietoja, erillisellä kirjautumissivulla, jossa on ruudut käyttäjätunnukselle ja salasanalle. Jos käyttäjä yrittää mennä suoraan mille tahansa muulle alasivulle kirjautumatta sisään, tulee sivuston ohjata käyttäjän kirjautumissivulle. Sivuston tulee myös jollain keinolla tarkkailla käyttäjän aktiivisuutta, jotta käyttäjä voidaan kirjata ulos liian pitkän epäaktiivisuuden takia.

6 TIETOTURVA

Tässä luvussa käydään läpi verkon tietoturva-asioista. Koska internet on hyvin turvaton paikka, ilman toimivaa suojausta saattaisi joku ulkopuolinen henkilö päästä käsiksi ohjaus ja monitorointi ominaisuuksiin, mistä voi seurata pahimmassa tapauksessa isoja vahinkoja. Jos dataa ei ole salattu millään tavoin, voisi joku ulkopuolinen henkilö mahdollisesti kaapata laitteiden välillä kulkevaa dataa tai jopa syöttää laitteille väärennettyä dataa mistä voi koitua isot vahingot. Tämän vuoksi on tärkeää olla olemassa mahdollisuus tehdä laitteisiin vähintään tietoturvapäivityksiä, että puutteet voidaan korjata ja mahdolliset vahingot minimoida.

6.1 Päivitykset

Palvelimella olisi hyvä olla päivityssovellus muita laitteita varten, jonka tehtävä on jakaa palvelimelle asennetut ohjelmistopäivitykset yhdistetyille laitteille. Päivitykset tulisi laittaa vain kerran yhdelle laitteelle, jonka jälkeen automaatio hoitaisi päivitykset muille laitteille.

Päivitysohjelmisto voisi toimia esimerkiksi siten, että kun palvelimelle on päivitetty jokin ohjelmisto, MQTT:n välityksellä lähetetään esimerkiksi aiheen ”update” alla viesti, joka sisältää päivitetyn ohjelmiston nimen ja versionumeron. Kaikki laitteet käsittelevät viestin, tarkastavat onko uusi versio eri kuin asennettu, asentavat tarpeen mukaan ja käynnistyvät uudestaan.

6.2 Salaus

Viestintä ja tietoliikenneprotokollat saattavat sisältää itsessään jonkin tasoisen salauksen, mutta yleensä saatetaan lisäksi käyttää esimerkiksi TLS:ää, joka salaa datan ja estää salakuuntelun.

6.3 Virheentarkistus

Projektiin sisältyvissä protokollissa on erilaisia virheentarkistusmenetelmiä, esimerkiksi TCP-protokollassa virheentarkistus ja korjaus saavutetaan käyttämällä tarkistussumma, kuittauksia ja aikakatkaisua.

Tarkistussumma on siirrettävästä datasta jollain algoritmilla luotava uniikki vakio-pituinen arvo, joka siirretään datan mukana vastaanottajalle. Vastaanottaja laskee vastaanotetusta tiedosta tarkistussumman, käyttäen samaa algoritmia ja vertaa sitä lähettäjän lähettämään tarkistussummaan. Tarkistussummien pitäisi täsmätä täysin, jos tarkistussummat eivät täsmää, vastaanottaja ilmoittaa tämän lähettäjälle ja lähettäjä toimittaa kyseisen kehyksen uudestaan. TCP-protokollan tarkistussummaa pidetään nykystandardeilla melko heikkona suojauksena. Sitä voidaan kompensoida käyttämällä esimerkiksi CRC-tarkistussummaa. /17/

Kuittauksia käytetään vastaanottaessa dataa. Kuittaus lähetetään viestin lähettäjälle, jolloin tämä tietää vastaanottajan olevan elossa ja yhteyden toimivan. Jos seuraavaa kehystä tai kuittausta ei tule määritetyn aikaikkunan sisällä, voidaan yhteys aikakatkaista virheiden välttämiseksi.

6.4 Käyttäjän todennus

On erittäin tärkeää vartioida sitä, kuka voi ottaa yhteyden palvelimeen ja kuka pääsee käsiksi käyttöliittymään. Käyttöliittymään täytyy toteuttaa jonkinlainen salasanan vaativa käyttäjän todennus, jolloin voidaan myös valvoa, kuka on tehnyt muutoksia järjestelmän kautta, jolloin väärinkäytön riski pienenee. Jos etäkäyttöliittymän käyttäminen vaatii VPN-yhteyden, on siinä jo yksi todennus. Itse käyttöliittymässä olisi kuitenkin oltava vielä oma todennus, että vain ihmiset jotka tietävät mitä tekevät ja jotka ovat oikeutettuja tekemään muutoksia, pääsevät niitä tekemään.

On myös tärkeää, että salasanat ja käyttäjätunnukset ovat salattuja, ettei kuka vain pääse luvatta näkemään niitä.

7 PROTOTYYPPI

Tässä luvussa käydään läpi prototyypin asennus, tarvittavan ohjelmiston luonti ja testaaminen.

7.1 Vaatimusmäärittely

Taulukossa 4 on listattuna järjestelmän vaatimukset, joiden prioriteetti on 1 (korkea) – 5 (matala).

Taulukko 4. Järjestelmän vaatimukset

ID	Prioriteetti	Vaatimus
1	1	Järjestelmän on toimittava, vaikka kaikilla laitteilla ei olisi kiinteitä IP-osoitteita
2	1	Järjestelmän tulee olla salasana suojattu
3	1	Viestiyhteyksien tulee olla salattuja
4	1	Paikallista käyttöliittymää tulee pystyä käyttää, vaikka ei olisi internet-yhteyttä
5	2	Käyttöliittymien tulee olla toiminnoiltaan identtiset
6	2	Käyttöliittymien tulee olla selainpohjaisia
7	3	Järjestelmän tulee tallentaa viestit tietokantaan
8	3	Järjestelmän tulee käyttää yrityksen omaa T-Plat.E sovelluskehitysalustaa
9	4	Järjestelmän tulee käyttää jotain julkista pilvipalvelua
10	5	Järjestelmän ohjelmistot tulee olla helposti päivitettävissä

7.2 Laitteisto

Prototyypilaitteistona käytetään Tietolaitteen omaa NXP:n LPC1769-prosessoriin perustuvaa sulautettujen järjestelmien kehitysalustaa, joka toimii IoT-laitteena. Tämä kehitysalusta valittiin projektiin ethernet-liitäntänsä vuoksi ja koska se oli heti saatavilla.

Testauksen alkuvaiheessa käytetään viestipalvelimenä Windows 10-tietokonetta, mutta projektin edettyä siirrytään käyttämään Amazonin AWS-pilvipalvelua.

Palvelinlaitteistona projektissa käytetään Amazonin AWS-palvelua, sen ilmaisen koeajan ja laajojen ominaisuuksien takia.

7.3 Ohjelmisto

IoT-laitteen ohjelmistona on käytössä Tietolaite Oy:n oma sulautettujen järjestelmien sovelluskehitysalusta T-Plat.E, joka on valikoitu sarja yleiskäyttöisiä ja keskenään yhteensopivia funktiolohkoja ja kirjastoja. T-Plat.E:n mukana tulee myös FreeRTOS reaaliaikakäyttöjärjestelmä. T-Plat.E:stä löytyy ethernet-yhteyttä varten avoimen lähdekoodin lwIP (lightweight IP) TCP/IP-pino. lwIP:n uudemmassa versiossa on ominaisuutena MQTT-protokolla, jonka integroin toimimaan käytössä olevan vanhemman lwIP-version kanssa. Prototyypivaiheen alussa, käytetään vain sisäverkkoa ja käytän omalla työasemallani Mosquitto Broker-viestinvälitysohjelmistoa.

Amazon Web Services-pilvipalveluun asennetaan myös Mosquitto Broker, joka asetetaan siltaan paikallisen Mosquitto Brokerin kanssa. Paikallinen välitysohjelmisto asetetaan tilaamaan sekä lähettämään kaikkia aiheita pilvipalvelussa olevan ohjelmiston kanssa. Tämä tehdään sen takia, että kaikki laitteet, sovellukset ja käyttöliittymät saavat aina tarvitsemansa tiedot viestinvälittäjiltä.

7.4 Asennus

Asennuksen aikana ei ilmennyt suurempia ongelmia, eniten aikaa meni MQTT-protokollan muokkaaminen toimivaksi vanhemman lwIP-version kanssa. IoT-laite ja siihen ajettu ohjelmisto testattiin toimivaksi käyttämällä samaan verkkoon kytkeytyllä Windows-tietokoneella Wireshark-pakettianalysaattoria ja Mosquitto Broker MQTT-välitysohjelmistoa.

Kun MQTT-viestit saatiin kulkemaan molempiin suuntiin, tehtiin käsittelyfunktiot tilatuille viestityypeille. IoT-laite myös lähettää nousevaa lukua aina, kun siihen

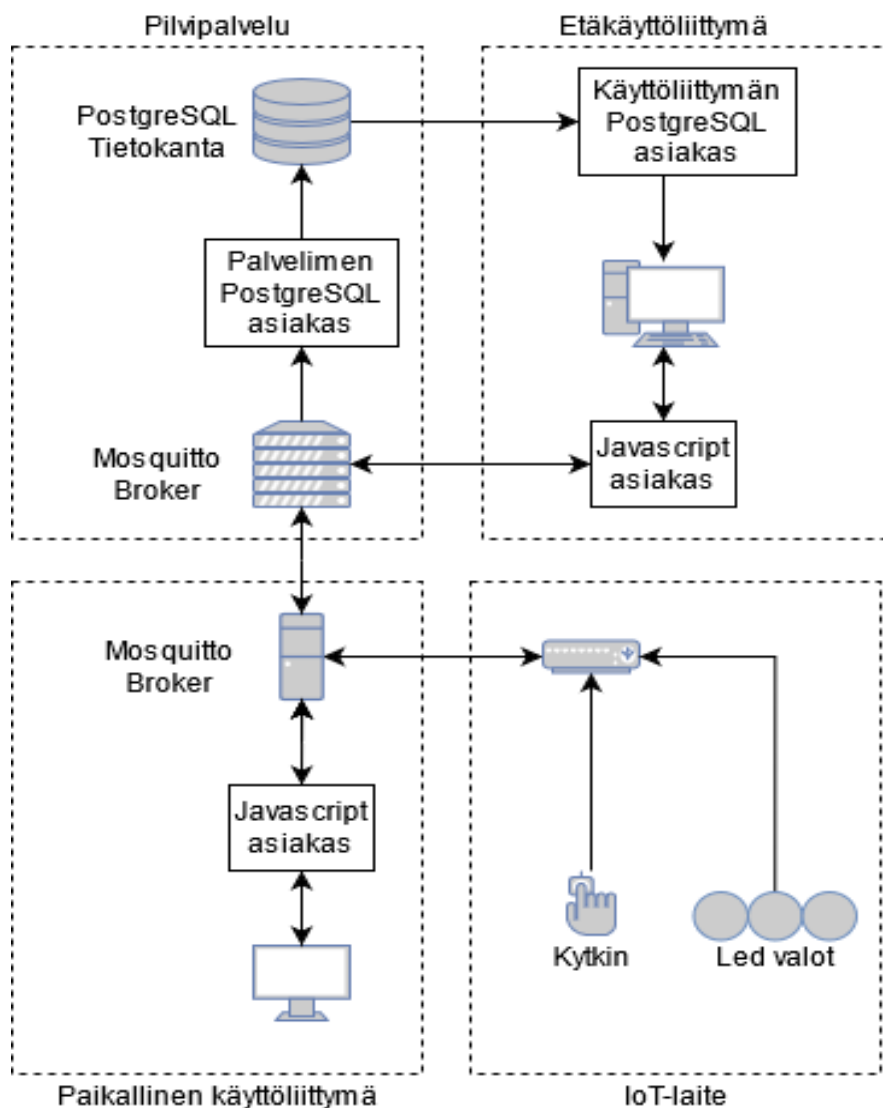
kytketyn kytkimen tila muuttuu, joten palvelin säädettiin lähettämään aiheen viimeisimmän arvon, aina kun joku tilaa aiheen. IoT-laitteelle tehtiin käsittelyfunktio, joka asettaa viimeisimmän arvon kytkinlaskurin lähtöarvoksi. Kuvassa 10 on Mosquitto Broker-ohjelmiston näkymä, kun IoT-laite ja paikallinen käyttöliittymä ovat avanneet yhteyden.

```
1619500934: mosquitto version 2.0.10 starting
1619500934: Config loaded from C:\users\tlmari\Desktop\mosquittoconf\mosquitto.conf.
1619500934: Opening ipv4 listen socket on port 1883.
1619500934: Opening websockets listen socket on port 9001.
1619500934: mosquitto version 2.0.10 running
1619500960: New connection from 192.168.21.125:4097 on port 1883.
1619500960: New client connected from 192.168.21.125:4097 as IoTDevice (p2, c1, k0, u'iotdev').
1619500960: No will message specified.
1619500960: Sending CONNACK to IoTDevice (0, 0)
1619500960: Received SUBSCRIBE from IoTDevice
1619500960:   iotethernet/ledblock/states (QoS 0)
1619500960: IoTDevice 0 iotethernet/ledblock/states
1619500960: Sending SUBACK to IoTDevice
1619500960: Received SUBSCRIBE from IoTDevice
1619500960:   iotethernet/switch/count (QoS 0)
1619500960: IoTDevice 0 iotethernet/switch/count
1619500960: Sending SUBACK to IoTDevice
1619500967: New client connected from 127.0.0.1:58956 as JavaScriptClient (p2, c1, k0, u'jsclient').
1619500967: No will message specified.
1619500967: Sending CONNACK to JavaScriptClient (0, 0)
1619500967: Received SUBSCRIBE from JavaScriptClient
1619500967:   iotethernet/switch/count (QoS 0)
1619500967: JavaScriptClient 0 iotethernet/switch/count
1619500967: Sending SUBACK to JavaScriptClient
1619500967: Received SUBSCRIBE from JavaScriptClient
1619500967:   iotethernet/ledblock/states (QoS 0)
1619500967: JavaScriptClient 0 iotethernet/ledblock/states
1619500967: Sending SUBACK to JavaScriptClient
```

Kuva 10. Mosquitto Broker ohjelmiston näkymä

7.4.1 Verkko

Laitteet on kytketty verkkoon seuraavalla tavalla: IoT-laite on ethernet verkkokaapelilla kytketty paikallisen käyttöliittymän tietokoneeseen. Paikallisen käyttöliittymän tietokoneelle asennettu Mosquitto Broker, on sillassa Amazonin palvelimella olevan Mosquitto Brokerin kanssa. Etäkäyttöliittymä ottaa yhteyden Amazonin palvelimella olevaan Mosquitto Brokeriin. Kuvassa 11 on järjestelmän arkkitehtuuri.

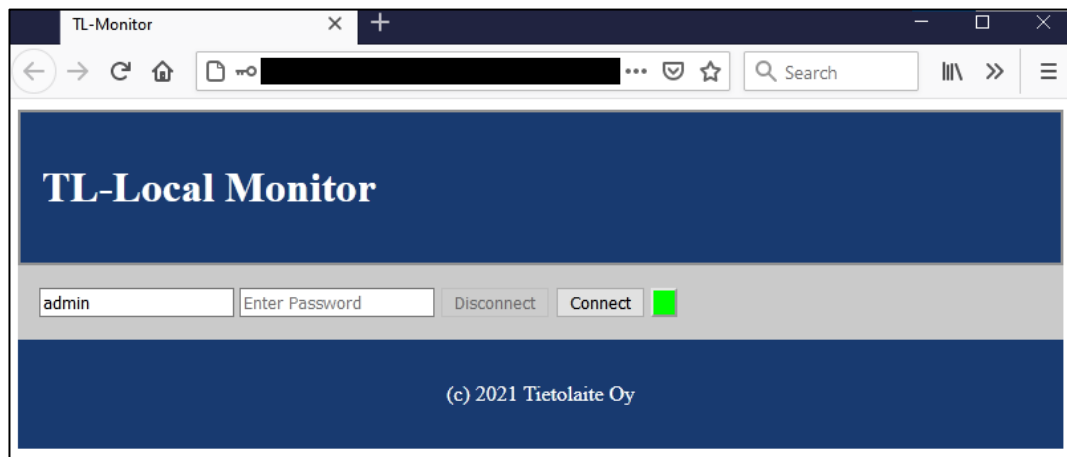


Kuva 11. Prototyypijärjestelmän arkkitehtuuri

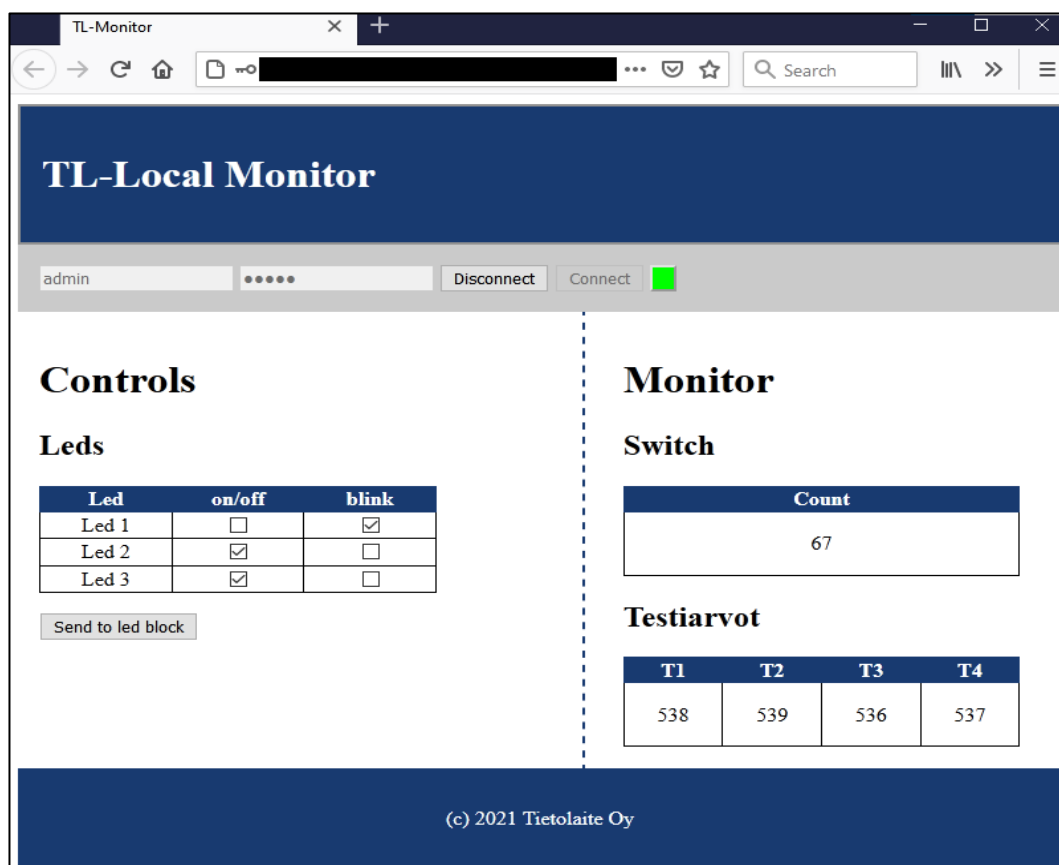
7.4.2 Paikallinen käyttöliittymä

Paikallinen käyttöliittymä on HTML:ään, CSS:ään ja JavaScriptiin perustuva paikallinen verkkosivu. Sivulla on kirjautumisominaisuus, manuaalinen yhdistäminen ja yhteyden katkaisu sekä yksi ohjausobjekti ja yksi valvontaobjekti. Sivun MQTT-ominaisuus perustuu Eclipsen Paho MQTT-ohjelmiston JavaScript-versioon ja se toimii käyttäen WebSocket teknologiaa. Sivusto voi tilata ja toimittaa haluttuja viestejä. Kuvassa 12 on sivuston aloitus näkymä, kun yhteyttä viestinvälitysohjelmistoon ei ole vielä luotu.

Koska tietoturva oli olennainen osa tätä opinnäytetyötä, on paikallinen käyttöliittymä suojattu paikallisella kirjautumisella. Kun sallittu käyttäjä on syöttänyt oikean käyttäjänimi ja salasana yhdistelmän, voi käyttäjä yhdistää välittäjäohjelmistoon painamalla ”Connect”-näppäintä. Tämän jälkeen aukeaa kuvan 13 mukainen näkymä, jossa ohjaus- ja valvontatoiminnot ovat käytettävissä.



Kuva 12. Paikallinen käyttöliittymä ennen välittäjään yhdistämistä



Kuva 13. Paikallinen käyttöliittymä kirjautumisen jälkeen

Tällä hetkellä sivulla on hallinnallisia ominaisuuksia kolme. Ohjaustoiminnoista löytyy ledien ohjaus, josta pystyy valintaruutuja käyttäen lähettää kehitysalustassa kiinni oleville kolmelle ledille ohjauskäskyjä, jotka ovat päälle, pois ja välkytys. Valvontatoiminnoista löytyy kytkin, jonka asennon muutoksia kerätään talteen kehitysalustalla. Luku kasvaa aina kun kytkimen asento muuttuu ja välittäjäohjelmisto pitää tallessa viimeisimmän arvon, vaikka IoT-laitteen sammuttaisi, jolloin tämä arvo voidaan palauttaa IoT-laitteelle ja jatkaa viimeisimmästä välittäjäohjelmistolle lähetetystä luvusta. Valvontaominaisuuksista löytyy myös testiarvot, joiden arvot kasvavat kaksi kertaa sekunnissa.

7.4.3 Amazon Web Services

Amazon Web Services asennettiin Amazonin blogista löytyvän ohjeen perusteella /18/. Tavoitteena oli käyttää Mosquitto Brokeria silta-tilassa, jolloin tietokoneella,

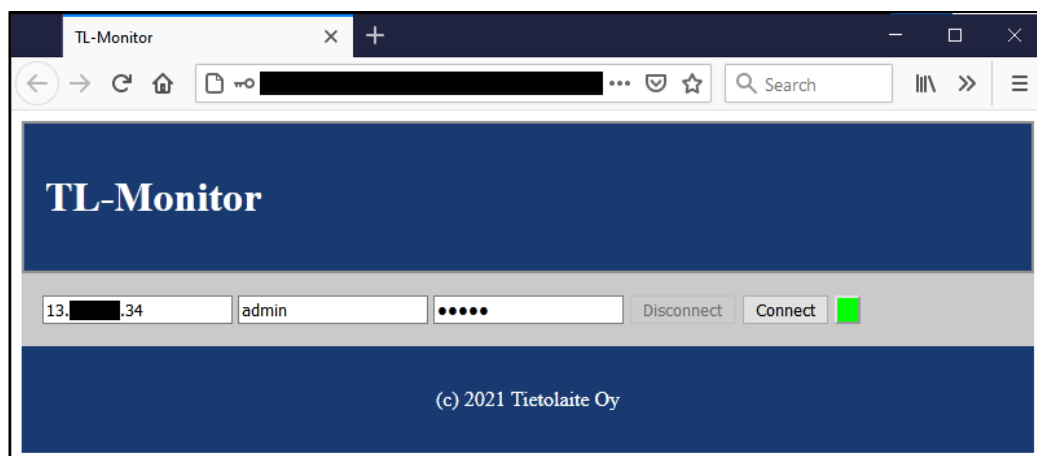
jolla paikallinen käyttöliittymä on, oleva Mosquitto Broker ja Amazonin palvelimella toimiva Mosquitto Broker toimittavat kaikki vastaanottamansa viestit toisilleen.

Palvelimen käyttöjärjestelmäksi valikoitui ohjeesta löytyvän Ubuntu Server 18.04 LTS:n sijaan Ubuntu Server 20.04 LTS, vanhemman version toimivuusongelmien takia. Palvelimen sijainniksi valittiin lyhimmän etäisyyden perusteella Tukholma, Ruotsi.

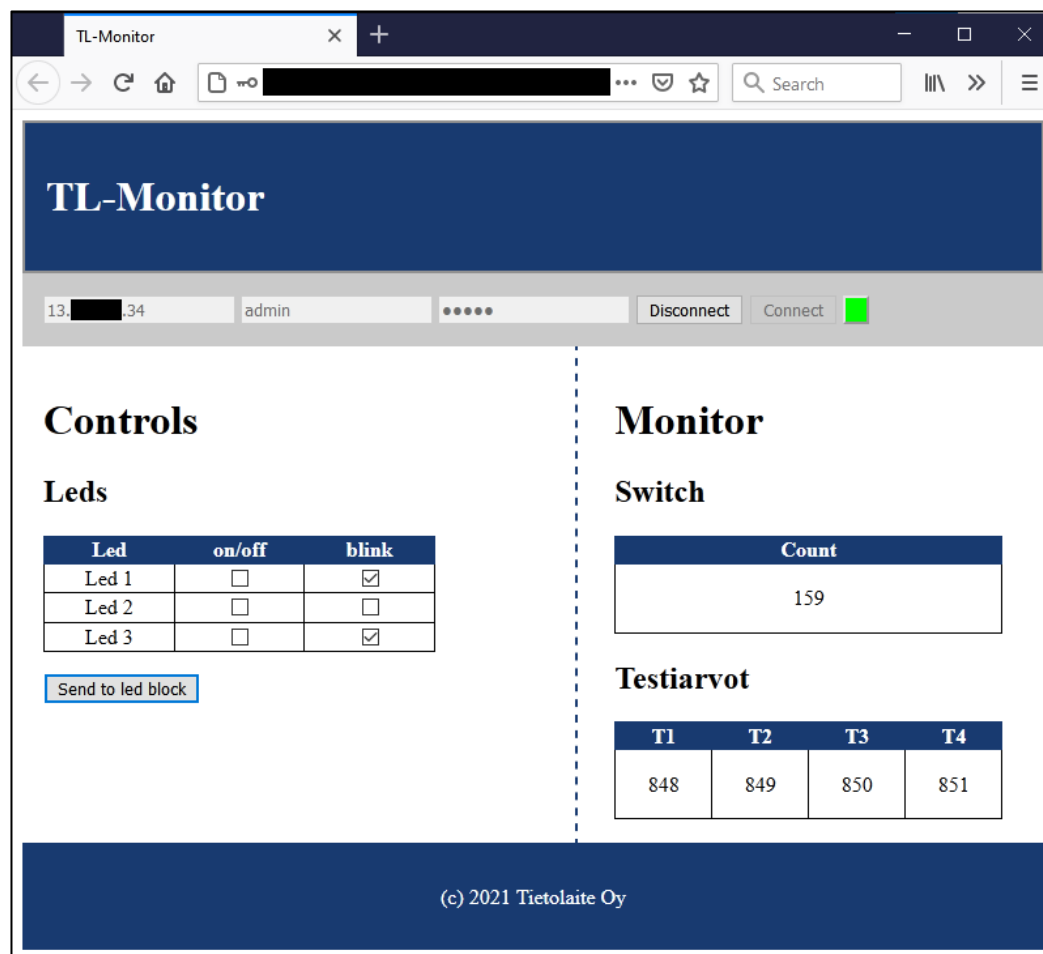
7.4.4 Etäkäyttöliittymä

Etäkäyttöliittymä on lähes identtinen paikallisen käyttöliittymän kanssa. Ainoana erona on, että etäkäyttöliittymä antaa käyttäjän asettaa MQTT palvelimen IP-osoitteen. Kuvassa 14 on etäkäyttöliittymä ennen välittäjään yhdistämistä ja kuvassa 15 etäkäyttöliittymä yhdistämisen jälkeen.

Etäkäyttöliittymään syötetään pilvipalvelussa toimivan Mosquitto Broker-välittäjäohjelmistoon asetetut käyttäjätunnus ja salasana. Välittäjäohjelmistoon voidaan asettaa monia käyttäjiä, jolloin jokaisella järjestelmää käyttävällä on omat kirjautumistunnukset. Tämä helpottaa käytön valvomista.



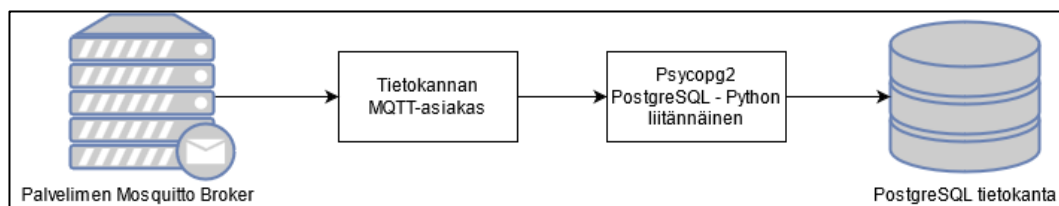
Kuva 14. Etäkäyttöliittymä ilman yhteyttä välittäjään



Kuva 15. Etäkäyttöliittymä yhdistettynä välitysohjelmistoon

7.4.5 Tietokanta

Tietokannaksi palvelimelle asennettiin PostgreSQL. Tietokantaa varten ohjelmoitiin, Python-ohjelmointikieltä käyttäen, palvelimelle oma MQTT asiakas, joka tilaa kaikki välittäjän aiheet ja tallentaa niihin tulevat viestit tietokannan tauluihin. Jos tietokannasta ei löydy saapuneen viestin aiheelle taulua, viestiä ei tallenneta tietokantaan. Tämä johtuu siitä, että jos ohjelma generoisi uuden taulun viestille, jolle ei ole omaa taulua, voisi vikatilanteessa generoitua todella iso määrä virheellisiä tauluja. Tauluun tallentuu tilatiedon lisäksi myös aikaleima ja automaattisesti kasvava järjestysluku. Palvelimelle tehdystä MQTT-asiakkaasta tehtiin palvelu, eli sitä ajetaan koko ajan palvelimen taustaprosessina. Kuvassa 16 tietokannan MQTT-asiakkaan arkkitehtuuri.



Kuva 16. Tietokannan MQTT-asiakkaan arkkitehtuuri

7.5 Testaus

Tässä kappaleessa käydään läpi testaamisen eri vaiheita ja menetelmiä. Tavoitteena testaamisessa on testata kaikkien verkon osien toiminta yksin, yhdistelminä ja kokonaisuutena.

7.5.1 Testisuunnitelma

Ensin verkon osille suoritetaan yksikkötestaus, jossa varmistetaan, että yksittäiset osat toimivat eri tilanteissa kuten pitää. Sitten suoritetaan integraatiotestaus, jossa testataan verkon osien välinen kommunikaatio, eli IoT-laitteelta paikalliseen käyttöliittymään ja viestinvälitysohjelmistoon, paikalliselta viestinvälitysohjelmistolta pilvipalvelun viestinvälitysohjelmistolle sekä pilvipalvelun viestinvälitysohjelmistolta etäkäyttöliittymälle ja tietokannan MQTT-asiakkaalle. Myös MQTT:n tietokannan välinen toimivuus testataan. Kun yksikkötestaus on suoritettu, suoritetaan vielä integraatiotestaus ja järjestelmätestaus, jossa tarkastellaan koko järjestelmän toimintaa.

Yksikkötestauksessa IoT-laitteelle saapuvien viestien käsittelyfunktio testataan syöttämällä sille oikeita ja vääriä arvoja. Saapuneiden viestien käsittelyfunktiolle annetaan viestejä, joissa on aihe, jolle ei ole määritetty mitään käsittelyä ja viestejä, joiden aiheessa on kirjoitusvirhe. Aiheiden käsittelyfunktioille annetaan viestejä joiden sisällöt on virheellisiä. Ledien ohjauskomentojen käsittelyfunktio ottaa vastaan kolmen numeron sarjan, jossa numeroiden arvot voivat olla 0 - led on pois päältä, 1 - led palaa koko ajan tai 2 - led välkky. Ledien ohjauskomentojen käsittelyfunktiolle syötetään testissä tyhjät arvot, vääriä numerosarjoja, liian pitkiä numerosarjoja, liian lyhyitä numerosarjoja, merkkejä sekä numeroita ja merkkejä se-

kaisin. Kytkimen tilanmuutoslaskurille tilauksen yhteydessä vastaanotettavan, viimeisimmän tilatiedon käsittelyfunktio testataan antamalla sille tyhjä viesti, negatiivinen luku, merkkejä sekä numeroita ja merkkejä sekaisin. Palvelimella olevaa tietokannan MQTT-asiakasta testataan syöttämällä sille viestejä, joiden aiheelle ei ole taulua tietokannassa, viestejä, joiden aiheen nimessä on pieniä virheitä. Tietokannan MQTT-asiakasta testataan lähettämällä sille viestejä, joiden viestisisältö ylittää asetetun sadan tavun pituusrajan.

Integraatiotestauksessa testataan, että kaikki verkon osat osaavat ottaa yhteyden toisiinsa ja että osien välinen kommunikaatio toimii ongelmitta. Yhteydet testataan siten, että lähetetään useita viestejä laitteelta toiselle tai sovelluksesta toiseen ja tarkastetaan että toinen laite on vastaanottanut kaikki sille lähetetyt viestit.

Järjestelmätestaus suoritetaan laittamalla IoT-laite lähettämään viestejä täydellä kapasiteetilla kymmenen minuutin ajan ja tarkastellaan siirrettyjen viestien määrää. Automaattilähetyksen ohella testataan myös kytkimien tilamuutosten lähetystä ja vastaanottokykyä. Tämän projektin välineistöllä ja ohjelmistoilla, tihein mahdollinen lähetystahti esitestausten perusteella, IoT-laitteelta palvelimelle, on noin kahdeksan automaattisesti lähetettyä viestiä sekunnissa, jotka lähetetään puolen sekunnin välein neljän viestin purkauksina. Laitteen pitäisi kymmenen minuutin aikana teoreettisesti lähettää 4800 viestiä.

7.5.2 Yksikkötestaus

IoT-laitteelle saapuneet virheelliset viestit hylättiin tässä projektissa. Normaalissa käytössä, tällaisessa tilanteessa asiasta pitäisi jollain tavoin ilmoittaa järjestelmävalvojalle. Ledien ohjauskomentojen käsittelyfunktio hylkäsi kaikki viestit, joiden sisältö ei ole kolmen numeron sarja tai sarjat, joissa olevien numeroiden sisältö ei ole 0, 1 tai 2. Kytkimen tilatietojen käsittelyfunktio hylkää virheellisen arvon ja asettaa laskurin arvoksi nollan.

Tietokannan MQTT-asiakas ei käsittele viestejä, joiden aiheessa on virhe, joiden aihetta ei löydy taulusta tai viestejä, joilla ei ole aihetta. Tietokannan MQTT-asiakas ei ole toistaiseksi ohjelmoitu tarkistamaan tietokantaan tallennettavan datan

oikeellisuutta, vaan jokaisesta viestistä tarkistetaan vain aihe ja viestisisällön koko. Koon tarkistusta testattiin syöttämällä ohjelmalle liian pitkiä viestejä, jotka tietokannan MQTT-asiakas jättää vastaanottaessaan käsittelemättä.

7.5.3 Integraatiotestaus

IoT-laitetta testattiin kytkemällä se paikallisen käyttöliittymän PC:n kanssa samaan verkkoon ja lähettämällä sille ping-käskyjä. Kun laite vastasi ping-käskyihin, säädettiin laite ottamaan yhteyden PC:llä ajettavaan Mosquitto Broker-viestinvälitys-ohjelmistoon ja tilaamaan testi aiheen. PC:llä olevalla Mosquitto Broker-ohjelmiston mukana tulleella Mosquitto Publish-ohjelmalla lähetettiin viestejä testi aiheeseen ja varmistettiin että ne välitetään IoT-laitteelle. Kun viestit kulkivat PC:ltä IoT-laitteelle, testattiin yhteys toisin päin. PC:llä laitettiin Mosquitto Broker-ohjelmiston mukana tullut Mosquitto Subscribe-ohjelma tilaamaan testi aihetta ja IoT-laite asetettiin lähettämään tasaisin väliajoin viestejä testi aiheeseen.

Paikallisen Mosquitto Broker-ohjelmiston ja pilvipalvelimelle asennetun Mosquitto Brokerin välinen silta testattiin lähettelemällä molempiin suuntiin useita viestejä uusista aiheista. Toimintalogikirjoista varmistettiin, että molemmat todella saavat automaattisesti kaikki toisen viestinvälittäjäohjelmiston vastaanottamat viestit.

Pilvipalvelun viestinvälitysohjelman ja tietokannan MQTT-asiakkaan välinen yhteys testattiin lähettämällä useita viestejä, asiakkaan tunnistamilla aiheilla. Lähetysten jälkeen tarkasteltiin tietokantojen sisältöä ja verrattiin sitä lähetyst historiaan.

7.5.4 Järjestelmätestaus

Testauksen perusteella IoT-laite kykenee hyvin yhtäjaksoiseen, 4 viestiä kerralla, lähetystahtiin. Tämän lisäksi IoT-laite kykenee automaattisten lähetystensä ohella ottamaan vastaan ja suorittamaan sille lähetetyt käskyt sekä puskuroimaan kytkimen tilatietoja, jotka se pystyy lähettämään automaattisten viestin lähetysten välissä. Tietokanta pysyy hyvin perässä, eikä hukkaa viestejä. Samoin käyttöliittymät pysyvät hyvin viestitahdin mukana. Testin päätteeksi tietokantaan oli tallennettu 4634 viestiä, joka on 96,5 % teoreettisesta maksimista mikä tarkoittaa, että järjestelmä on hyvin lähellä optimia. Ero teoreettisen määrän ja todellisen määrän välillä

selittyy todennäköisesti sillä, että automaattisten testiviestien lähetyksen laukaisu jättää ajallisesti jonkin verran.

8 JOHTOPÄÄTÖKSET

MQTT on viestintäprotokollana erinomainen ja helppokäyttöinen. Se on myös tehokas sen keveyden huomioon ottaen. Mosquitto Broker-viestinvälitysohjelmia voidaan vaivatta asettaa siltaan toistensa kanssa, jolloin voidaan paikallinen MQTT:tä käyttävä verkko yhdistää helposti palvelimella toimivaan palvelimeen. Tällä tavoin voidaan toteuttaa helposti paikallinen käyttöliittymä ja etäkäyttöliittymä, joissa on täysin samat ohjaus- ja monitorointiominaisuudet. Tämä tapa rakentaa MQTT-verkko on myös hyvä sen takia, että jos jostain syystä nettiyhteys katkeaa tai etäpalvelimeen tulee ongelmia, voidaan järjestelmää ajaa normaalisti paikallisesta käyttöliittymästä, sillä palvelimella oleva ja paikallinen viestinvälitysohjelma vain jakavat kaikki viestit toisilleen.

Amazon Web Services-palvelu oli helppokäyttöinen, joskin sisältää jonkin verran toimivuus ongelmia. Heidän omat ohjeensa eivät ole täysin ajan tasalla, mikä vaikeuttaa paljon palveluiden käyttöönotossa. Itse palvelin toimi erinomaisesti kun se oli säädetty oikein, mutta AWS:n tarjoamia IoT- ja tietokantapalveluita en saanut tämän opinnäytetyön aikana toimimaan kunnolla, mistä johtuen jouduin itse lisäämään ja tekemään palveluiden osat suoraan palvelimen Ubuntu-käyttöjärjestelmälle.

Tietoturvaa tällaisessa verkossa tarjoavat useat protokollat ja salasanasuojaukset. MQTT-viestintäprotokolla sisältää itsessään salasanasuojauksen ja viestin sisällön salauksen, jonka lisäksi asetin etäkäyttöliittymään ja paikalliseen käyttöliittymään vielä erilliset käyttäjätilit ja salasانات tuomaan lisää suojausta.

Käyttöliittymien tekeminen oli ongelmaton, sillä ne ovat vain yksinkertaiset HTML-, JavaScript- ja CSS-pohjaiset verkkosivut. Eniten huomiointia käyttöliittymissä vaati JavaScriptiin ja WebSocketeihin perustuvan Eclipsen Paho MQTT-asiakas ohjelman säätäminen ja kehittäminen opinnäytetyön vaatimusten tasolle.

Projektin vaatimuksista kaikki paitsi päivitysominaisuus täyttyi. Päivitys ominaisuudesta ei tässä opinnäytetyössä sen laajuuden vuoksi tehty muuta kuin alustava suunnitelma.

LÄHTEET

- /1/ ARM Limited. Arm verkkosivujen sanasto. Viitattu 6.5.2021.
<https://www.arm.com/glossary/iot-devices>
- /2/ Arrow Electronics, Inc. IoT Operating Systems. Viitattu 6.5.2021.
<https://www.arrow.com/en/research-and-events/articles/iot-operating-systems>
- /3/ Techfocus media, Inc. All About Messaging Protocols. Viitattu 6.5.2021.
<https://www.eejournal.com/article/20150420-protocols/>
- /4/ MQTT Organization. MQTT:n viralliset verkkosivut. Viitattu 20.1.2021.
<https://mqtt.org/>
- /5/ Object Computing. DDS:n verkkosivut. Viitattu 21.1.2021.
<https://DDS.org/>
- /6/ OASIS Open. AMPQ:n verkkosivut. Viitattu 21.1.2021.
<https://www.ampq.org/>
- /7/ The XMPP Standards Foundation. XMPP:n verkkosivut. Viitattu 25.1.2021.
<https://xmpp.org/>
- /8/ Internet Engineering Task Force (IETF). A Roadmap for Transmission Control Protocol (TCP) Specification Documents. Viitattu 25.2.2021.
<https://tools.ietf.org/html/rfc7414>
- /9/ Postel, J. User Datagram Protocol. Viitattu 25.2.2021.
<https://tools.ietf.org/html/rfc768>
- /10/ AO Kaspersky Lab. What is VPN? How It Works, Types of VPN. Viitattu 6.5.2021. <https://www.kaspersky.com/resource-center/definitions/what-is-a-vpn>
- /11/ Amazon Web Services, Inc. AWS:n verkkosivut. Viitattu 13.4.2021.
<https://aws.amazon.com/>
- /12/ Microsoft. Azuren verkkosivut. Viitattu 13.4.2021.
<https://azure.microsoft.com/>
- /13/ Google LLC. Google Cloudin verkkosivut. Viitattu 13.4.2021.
<https://cloud.google.com/>
- /14/ Hipp, D. R. SQLiten verkkosivut. Viitattu 5.2.2021 <https://www.sqlite.org/>
- /15/ Oracle Corporation. MySQL:n verkkosivut. Viitattu 8.2.2021.
<https://www.mysql.com/>

/16/ The PostgreSQL Global Development Group. PostgreSQL:n verkkosivut. Viitattu 8.2.2021.

<https://www.postgresql.org/>

/17/ Jones, E. 2008. TCP Checksums Are Not Enough. Viitattu 18.2.2021.

<https://www.evanjones.ca/tcp-checksums.html>

/18/ Amazon Web Services, Inc. AWS Blogikirjoitus. Viitattu 28.04.2021.

<https://aws.amazon.com/blogs/iot/how-to-bridge-mosquitto-mqtt-broker-to-aws-iot/>