Osaamista
ja oivallusta
tulevaisuuden
tekemiseen

Sakari Hakala

# Pilvipalvelun käyttö kaluston toiminnan visualisointiin

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Sähkö- ja automaatiotekniikka

Insinöörityö

3.3.2021

| | |
|---|---|
| Tekijä<br>Otsikko<br><br>Sivumäärä<br>Aika | Sakari Hakala<br>Pilvipalvelun käyttö kaluston toiminnan visualisointiin<br><br>36 sivua<br>3.3.2021 |
| Tutkinto | insinööri (AMK) |
| Tutkinto-ohjelma | Sähkö- ja automaatiotekniikka |
| Ammatillinen pääaine | Automaatiotekniikka |
| Ohjaajat | Teknologiavastaava Heikki Hakala<br>Yliopettaja Erkki Räsänen |


Insinöörityön aiheena oli pilvipalvelun käyttö kalustonhallintajärjestelmässä tiedon visualisointiin. Työn tavoitteena oli suunnitella järjestelmän rakenne ja valita pilvipalveluntarjoaja ja tarvittavat palvelut saatujen vaatimusten mukaisesti. Osa järjestelmän vaatimuksista tuli jo käytettävän laitteiston asettamana ja osa asiakkaan pyytämänä.

Tärkeimpänä laitteena järjestelmässä oli tarkoitus toimia EPEC 6107 -ohjainlaite, mutta ohjelmien asennusongelmien takia sitä ei saatu lähettämään dataa pilvipalveluun. Pilvipalvelusta data siirtyi tallennustilaan, josta nettisivua ylläpitävä virtuaalikone voi lukea datan ja esittää sen graafisessa muodossa. Myös nettisivun toteutus jäi työn ulkopuolelle, mutta dataa saatiin luettua tallennustilasta virtuaalikoneelle.

Pilvipalveluntarjoajan valinnassa oli tarkoituksena tehdä valinta ominaisuuksien ja hinnan perusteella, mutta ominaisuudet olivat tärkeämmässä osassa. Parhaaksi valittiin Microsoft Azure, koska sen ominaisuudet vastasivat vaatimuksia paremmin kuin Amazonin IoT-palvelu, ja hinta oli merkittävästi halvempi kuin Googlen palvelulla.

Työssä perehdyttiin myös yleisesti IoT-järjestelmän perusteisiin sekä tietoturvaan, jotta saataisiin parempi käsitys järjestelmän toiminnasta. Tietoturvan osalta verrattiin pilvipalvelua paikalliseen palvelimeen ja päädyttiin tulokseen, että pilvipalvelu on todennäköisesti turvallisempi ja edullisempi vaihtoehto.

Tarkoituksena oli myös kehittää prototyyppi kalustonhallintajärjestelmästä. Vaatimusten mukaista prototyyppiä ei saatu tehtyä, mutta työstä saatiin silti arvokasta tietoa järjestelmän mahdollisesta rakenteesta, ja sen perusteella voidaan tehdä arvio tarvittavasta työmäärästä järjestelmän kehittämiseksi. Tietoa saatiin myös siitä, mihin asioihin tarvittaisiin lisää ammattitaitoa, jotta järjestelmästä kehitettäisiin järkevä ja vakaa.




| | |
|---|---|
| Avainsanat | IoT, esineiden internet, kalustonhallintajärjestelmä |

| | |
|---|---|
| Author | Sakari Hakala |
| Title | Use of Cloud Services for Fleet Management Data Visualization |
| Number of Pages | 36 pages |
| Date | 3 March 2021 |
| Degree | Bachelor of Engineering |
| Degree Programme | Electrical and Automation Engineering |
| Professional Major | Automation Engineering |
| Instructors | Heikki Hakala, Chief Technology Officer<br>Erkki Räsänen, Principal Lecturer |

The subject of the thesis is the utilization of cloud services for fleet management and visualization of data. The goal was to design the system and choose a cloud service provider that can fulfill the requirements. The client set the requirements, which were affected by the used hardware.

The intended logic controller for the system was EPEC 6107 display unit but due to issues with installing the needed software, the data transfer from it to the cloud was not successful. The data was transferred from the cloud service to a storage account, from which a virtual machine can access it. The virtual machine also hosts a website and there the user can see the data in graphs or lists. The realization of the website was left outside of the thesis project's scope, but virtual machine was able to read the data from storage service.

The cloud service provider was chosen by comparing features and price, but the needed features had to be included. Microsoft Azure was chosen because its features were closer to requirements than Amazon's cloud service's and price was significantly lower than Google's.

The thesis also clarifies the basics of IoT systems and security in cloud, to get a better understanding of the system and how it works. In security section, cloud service and local server were compared, and cloud service was deemed the more secure and cheaper option.

Another goal was to implement a prototype of the fleet management system. This goal was not reached but the thesis work gave valuable information about a possible structure of the system and it may help with estimating the workload, if the system will be developed. Information about what expertise would be needed to make a stable system was also received.

| | |
|---|---|
| Keywords | IoT, Internet Of Things, Fleet management |

## Sisällys

## Lyhenteet

| | |
|---|---|
| GiB | Gibitavu. Tallennustilan mittayksikkö, joka on noin 1,07 GB. |
| IoT | *Internet of Things.* Esineiden internet. |
| JSON | *Javascript Object Notation.* Tekstipohjainen tiedostomuoto tiedonlähetykseen ja tallennukseen. |
| LRS | *Locally redundant storage.* Microsoft Azuren datan tallennusmenetelmä, jossa data on kopioitu kolmesti samaan fyysiseen sijaintiin. |
| MQTT | *Message Queuing Telemetry Transport.* Viestintäprotokolla pienelle datamäärälle |
| SSL | *Secure Sockets Layer.* Salausprotokolla viestintään verkossa. |
| TLS | *Transport Layer Security.* Salausprotokolla viestintään verkossa. SSL-protokollan päivitetty versio |

# 1 Johdanto

Insinöörityön tavoitteena on suunnitella kalustonhallintajärjestelmä, jonka laitteet lähettävät tietoja pilvipalveluun ja josta niitä voidaan katsoa. Työssä verrataan eri pilvipalvelutarjoajien hintoja ja palvelujen ominaisuuksia ja valitaan niistä toteutukseen parhaiten soveltuvat palvelut. Työssä tehdään myös prototyyppi, jolla kokeillaan suunnitelman toimivuutta pilvipalveluiden osalta.

Työ tehdään Hevtec Oy:lle, joka on sähkö- ja hybridijärjestelmien suunnitteluun erikoistunut insinööritoimisto. Yritys keskittyy pääasiassa suunnittelemaan uusien koneiden sähköjärjestelmiä ja vanhojen polttomoottorikoneiden muuntamista sähköisiksi tai hybrideiksi. Yrityksen toiminta painottuu työkoneisiin, ja kalustonhallintajärjestelmälle olisi kysyntää asiakkaiden, eli konerakentajien, puolelta. Tämä työ antaa tilaajalle tietoa, onko järjestelmän kehittäminen itse järkevä ratkaisu vai kannattaako palvelu ostaa valmiina.

Järjestelmän on tarkoitus auttaa koneen suunnittelevaa yritystä helpottamalla laitteen toiminnan seuraamista sen kehitysvaiheessa varsinkin vikatietojen, mutta muidenkin tärkeiden tietojen, kuten akkujännitteen ja -varauksen, avulla. Näiden tietojen avulla toimintaa voidaan muuttaa ja optimoida. Myös koneen rakentajaa tai myyjää järjestelmä helpottaa mahdollistamalla etäpäivitykset ohjainlaitteille.

Yritys vaatii järjestelmältä

- tietojen lähettämistä pilveen yhden minuutin välein, kun kone on käynnissä
- tietojen tallennusta 12 kk:n ajaksi
- tietojen näkymistä nettisivun kautta visuaalisesti
- mahdollisuutta lähettää parametreja ja tiedostoja laitteelle nettisivun kautta
- mahdollisuutta ladata valittu data nettisivulta listana
- mahdollisuutta lisätä erilaisia laitteita järjestelmään

- että visualisoinnit ovat laitekohtaisia
- mahdollisuutta hallita ja lisätä käyttäjiä, joilla on erilaisia oikeuksia katsoa tietoja.

Järjestelmä siis vaatii kaksisuuntaista liikennettä pilvipalvelun ja laitteen välille, ja tavan, jolla visualisointeja voidaan tehdä ja näyttää.

Työssä perehdytään ensin yleisesti IoT-järjestelmiin (Internet Of Things), jotka ovat laitteistoja, jonka yksi tai useampi laite on liitetty internetiin. Sen jälkeen perehdytään työn aiheena olevan järjestelmän laitteisiin ja suunnitellaan ratkaisun rakennetta ja tarvittavia osia. Sitä seuraavassa osiossa annetaan vaihtoehtoja tarvittaville palveluille kolmelta isolta pilvipalveluntarjoajalta, jotka ovat Google Cloud, Azure Web Services ja Microsoft Azure. Näille palveluille lasketaan hinnat, ja niistä valitaan sopivin hinnan ja ominaisuuksien perusteella. Tämän jälkeen esitetään järjestelmän toteutus ja viimeisenä perehdytään vielä yleisesti pilvipalveluiden tietoturvallisuuteen.

## 2 IoT-järjestelmän osat

IoT-järjestelmät ovat järjestelmiä, joissa laitteet lähettävät dataa internetin avulla datakeskuksiin tai vastaanottavat dataa datakeskuksista. Se mahdollistaa

- datan seuraamisen missä tahansa
- laitteiden välisen kommunikaation datakeskusten kautta
- datan yhdistämisen monelta laitteelta ja sen analysoinnin
- datan käsittelyn, vaikka tarvittaisiin suurtakin laskentatehoa
- datan pitkä- tai lyhytaikaisen tallentamisen
- laitteiden päivittämisen
- parametrien asetuksen laitteelle,

ilman että tarvitsee olla laitteen lähellä.

## 2.1 Reunalaskenta

Data, jota halutaan siirtää ja käsitellä, saadaan usein kenttälaitteilta. Useimmat sensorit ja laitteet eivät suoraan pysty liittymään internetiin, koska siihen tarvitaan enemmän laskentatehoa, muistia ja sähkövirtaa kuin on järkevää asentaa yksittäiseen anturiin. Tämän takia IoT-järjestelmät usein vaativat kenttälaitteiden ja internetin välille laitteen, joka mahdollistaa yhteyden. Tähän käytetään yhdyskäytäviä ja reunalaitteita. Reunalaite on laite, johon kenttälaitteet yhdistyvät, esimerkiksi Bluetooth- tai Wlan-tekniikoiden, jonkun muun langattoman tai langallisen väylätekniikan tai analogisignaalin avulla. Reunalaite voi mahdollisesti suodattaa, tai muuten käsitellä tietoa, jolloin puhutaan reunalaskennasta. (1.)

Reunalaitteella voi olla muitakin tehtäviä, kuten ratkaista reaaliajassa turvallisuuteen liittyvien tapahtumien toimenpiteet tai muuten ohjata liitettyjä kenttälaitteita. Käytettävä laite valitaan tehtävien vaatimusten mukaan. Jos laskentatehoa vaativia tehtäviä ei ole, voidaan selvitä halvemmalla ja pienemmän sähkönkulutuksen omaavalla laitteella. Toisaalta, jos laite esimerkiksi yhdistää tai pakkaa tietoa, voidaan säästää tiedonsiirtokustannuksissa, vaikka sähkönkulutus ja hankintakustannukset olisivatkin suuremmat. (1.)

Yleisesti reunalaitteen tehtäviä ovat

- reaaliaikaiset ohjaukset kenttälaitteille
- datan vastaanottaminen sensoreilta ja muilta liitetyiltä laitteilta
- internetyhteyden muodostaminen
- yhteyden muodostaminen pilvipalveluun
- datan säilöminen internetyhteyden katketessa
- datan salaaminen ja kokoaminen internetiin lähetettävään muotoon

- datan lähettäminen.

Tehtäviä voi olla myös enemmän tai vähemmän riippuen suunnitellusta järjestelmästä.

## 2.2 Pilvipalvelu

Toinen osuus järjestelmästä on pilvipalvelussa tapahtuva toiminta. Sen tehtäviä ovat yhdistettävän laitteen todennus, laitteen lähettämän datan vastaanottaminen, hallinta ja tallennus useilta eri laitteilta sekä mahdollisesti datan lähettäminen laitteille, esimerkiksi päivitysten tai parametrien muodossa. Pilvipalvelussa dataa voidaan myös analysoida tai esittää käyttäjälle.

Ominaista pilvipalvelun osuudelle järjestelmästä on hitaampi reagointi muutoksiin kuin reunalaskennalla. Tämän takia nopeaa ohjausta vaativat toimet suoritetaan reunalaitteella. Pilvessä voidaan myös asentaa toimintoja suoritettavaksi vastaanotetun datan perusteella käyttäen sääntökoneita, esimerkiksi lämpötilan noustessa raja-arvon yläpuolelle pilvipalvelu voi lähettää tekstiviestin.

## 2.3 IoT-järjestelmän suunnittelu

IoT-järjestelmässä on käytössä useita eri tekniikoita, joten sellaisen suunnitteluun tarvitaan tietoa useilta aloilta, joita ovat

- laitteistosuunnittelu
- virranhallinta
- sulautetut järjestelmät
- ohjelmointi
- kommunikaatiojärjestelmät ja -protokollat

- verkkoprotokollat
- tietoturva
- pilvipalvelut
- data-analytiikka, datan hallinta ja datatiede. (1.)

Järjestelmää suunnitellessa tiedossa on usein ominaisuus, jota pidetään tärkeänä järjestelmässä, esimerkiksi virranhallinta tai koko. Suunnittelijan täytyy ymmärtää käytetyn näkökannan vaikutukset järjestelmään. Jos esimerkiksi keskitytään vahvasti virranhallintaan, järjestelmän muut ominaisuudet jäävät heikommiksi. Suuremman laskentatehon laitteet käyttävät yleisesti enemmän energiaa. (1.)

Valittavia viestintätekniikoita, IoT-palveluita ja rajapintoja on useita satoja erilaisia, ja niissä on eroja skaalautuvuuden, käyttömukavuuden, nopeuden sekä hinnan osalta. Toisen verkkoprotokollan valitseminen voi aiheuttaa ongelmia jossain päin maailmaa. Suunnittelijan tehtävänä on selvittää ja antaa ratkaisuja koko järjestelmän kattaviin ongelmiin, kuten

- datan pääsy internetiin yhteyden rakoillessa
- kuinka vaarallista on datan menettäminen, ja missä sitä pitäisi hallita
- halutaanko jossain vaiheessa vaihtaa pilvipalveluntarjoajaa.

Nämä kaikki asiat vaikuttavat siihen, mitä protokollia valitaan ja missä dataa käsitellään ensimmäisenä. Jos kaiken datan saaminen pilveen on tärkeää, se täytyy ensin säilöä reunalaitteella, joka voi lähettää sen uudestaan myöhemmin, jos data ei saapunutkaan perille. Reunan lähellä tapahtuva datankäsittely voi myös pienentää internettiin lähetettävän datan määrää ja samalla datan siirrosta aiheutuvia kuluja. (1.)

Ensimmäisenä ja tärkeimpänä asiana suunnittelussa on se, mitä lisäarvoa esineiden internet antaa järjestelmälle (1). Laitetta ei kannata liittää internetiin, jos siitä saatavat hyödyt eivät ylitä kustannuksia. Tässä täytyy ottaa huomioon kiinteiden kustannusten lisäksi

kehityskustannukset, jotka todennäköisesti ovat suhteessa suuremmat, varsinkin jos asiakasmäärät eivät nouse korkeiksi. Kustannuksia verrataan saavutettaviin hyötyihin, esimerkiksi ongelmien selvitys ja ohjainlaitteen päivitys etänä sekä koneen toiminnan seuraaminen. Jos hyödyt ovat niin suuria, että ne kuittaavat arvioidut kehityskustannukset, kannattaa järjestelmän kehittämistä alkaa suunnittelemaan tarkemmin. Jos IoT tarjotaan lisäpalveluna järjestelmälle, täytyy myös arvioida sen hinta myyjän sekä ostajan näkökulmasta.

# 3 Tarvittavat sovellukset ja laitteet

Luvussa tutustutaan kalustonhallintajärjestelmän toteuttamiseen mahdollisesti käytettäviin komponentteihin. Tämä kokoonpano on yksi mahdollinen vaihtoehto. Ohjelmoitava logiikka oli työn tilaajan valitsema ja muut komponentit ovat palveluita ja sovelluksia, joita voidaan käyttää järjestelmän toteuttamiseksi.

## 3.1 Ohjelmoitava logiikka

Projektissa käytetään EPEC 6107 -ohjainlaitetta. Se on näytöllinen ohjelmoitava logiikka, jossa Codesys-suoritusympäristö suoritetaan Linux-käyttöjärjestelmässä. Ohjainlaitteella on valmiit toiminnallisuudet Internet-yhteyden muodostamiseen Ethernet-kaapelin ja mobiiliverkon avulla.

EPEC 6107 -laitteessa käytetty Linux on valmistajan oma jakelu, joten tarvittavien ohjelmien asennuksessa voi tulla ongelmia. Esimerkiksi Linuxille tyypillisiä paketinhallintasovelluksia ei ole valmiiksi asennettuna, mikä vaikeuttaa tarvittavien ohjelmien asentamista. Toinen mahdollinen ongelma saattaa olla muuttujien arvojen siirto Codesys-suoritusympäristöstä Linuxissa pyörivän ohjelman puolelle.

## 3.2 IoT-asiakassovellus (IoT client)

Ohjainlaitteelle täytyy asentaa tai kehittää pilvipalvelun IoT-palvelun asiakassovellus, joka hallitsee yhteyttä pilven IoT-keskukseen. Kaikilla tutkituilla pilvipalveluilla on useita eri vaihtoehtoja toiminnallisuuden suorittamiseen. Ohjelmointikielen lisäksi kehittäjän on

valittava, käytetäänkö pilvipalveluntarjoajan kehittämiä ohjelmistokehityspaketteja vai kehitetäänkö ratkaisu kokonaan itse. Ohjelmistokehityspaketit ovat ladattavia ohjelmointia helpottavia työkaluja, joissa on mahdollisesti ohjelmakirjastoja, dokumentointi ja ohjeet käyttöön.

Tutkitut pilvipalvelut käyttävät MQTT- ja TLS/SSL-protokollia viestimiseen. MQTT on protokolla, jota käytetään kahden laitteen välisen yhteyden muodostamiseen. SSL ja TLS ovat tietoliikenteen salaukseen käytettyjä protokollia, joista TLS on SSL-protokollan päivitetty versio. Näiden avulla voidaan ottaa yhteys ja lähettää dataa pilvipalveluihin, mutta tarvittava ohjelmointimäärä on suuri.

Ohjelmistokehityspakettien ohjelmointikieliksi voidaan valita mm. Java, NodeJs, Python tai C. Pilvipalveluntarjoajien kehittämät paketit helpottavat ohjelmointitaakkaa sisältämällä ohjelmakirjastoja tarvittavista toiminnoista. Esimerkiksi viestien salauksen ja lähettämisen jokaista tehtävää ei tarvitse itse kehittää. Kehittäjän tehtäväksi jää viestien kokoaminen haluamaansa muotoon ja ominaisuuksien yhdistäminen kokonaisuudeksi. Paketit tarvitsevat toimiakseen valitun ohjelmointikielen kääntäjän sekä paketin käyttämät yleiset ohjelmistopaketit, kuten tarvittavat MQTT- ja SSL-toiminnot.

### 3.3 IoT-hallintapalvelu

IoT-hallintapalvelu on pilven palvelu, joka nimensä mukaisesti hallitsee laitteiden kommunikaation pilvipalveluun. Sen tehtäviin kuuluu laitteiden yhdistäminen ja todennus, datan vastaanottaminen ja välittäminen eteenpäin sekä datan lähettäminen laitteille. Yhdistettävän laitteen todennus tehdään hallintapalvelussa luodun sertifikaatin tai salasanan ja tunnuksen avulla. Hallintapalvelussa laitteet jaetaan ryhmiin, jolloin esimerkiksi päivitysten lähettäminen onnistuu kerralla useammalle laitteelle. Hallintapalvelun päätehtävänä on siis siirtää data yksilöitynä laitteilta pilven tallennustiloihin.

### 3.4 Palvelin verkkosivulle

Järjestelmälle tarvitaan verkkosivusto, josta käyttäjä voi seurata koneiden lähettämää dataa ja lähettää koneelle tiedostoja. Verkkosivu tarvitsee toimiakseen palvelimen. Mah-

dollisuutena on ottaa palvelin samalta pilvipalveluntarjoajalta kuin IoT-palvelu, jolloin vältetään ylimääräiset palvelut tiedon siirtämiseen sekä tiedonsiirtomaksut, joita maksetaan vain pilvestä ulospäin tapahtuvasta tietoliikenteestä. Mahdollisuutena on myös lähettää data IoT-keskuksesta toiselle palveluntarjoajalle ja ylläpitää verkkosivustoa siellä.

Verkkosivun ylläpitoon on kaksi eri vaihtoehtoa. Ensimmäinen vaihtoehto on virtuaalikone, jossa palveluna on vain resursseja: keskusmuisti, laskentateho ja tallennustila. Silloin kehittäjä asentaa itse tarvittavat ohjelmat ja suoritusympäristöt verkkosivun toimimiseen. Tällöin käyttöjärjestelmän ja muiden ohjelmien päivitys jää kehittäjän vastuulle. Toisena vaihtoehtona on korkeamman tason pilvipalvelu, jossa käyttöjärjestelmä ja suoritusympäristöt ovat pilvipalvelun vastuulla ja asennettuna, jolloin kehittäjän vastuulle jää vain sovelluksen ja datan hallinta.

Virtuaalikoneen hyötyjä ovat edullisuus, skaalautuvuus ja joustavuus verrattuna korkeamman tason ratkaisuun, mutta kehittäjän vastuu jää suuremmaksi ylläpidon osalta. Projektissa käytetään virtuaalikonetta joustavuuden ja hinnan takia, sillä eri palveluiden yhdistämiseen voidaan tarvita erilaisia sovelluksia, joiden käyttäminen on mahdollisesti helpompaa joustavammassa palvelussa.

Palvelimen lisäksi tarvitaan tallennustila laitteiden lähettämälle datalle. Palvelimella on itsessään muistia, mutta tarvitaan myös ulkoinen tallennustila laitteiden pilvipalveluun lähettämän datan takia.

### 3.5 Palvelu visualisointiin

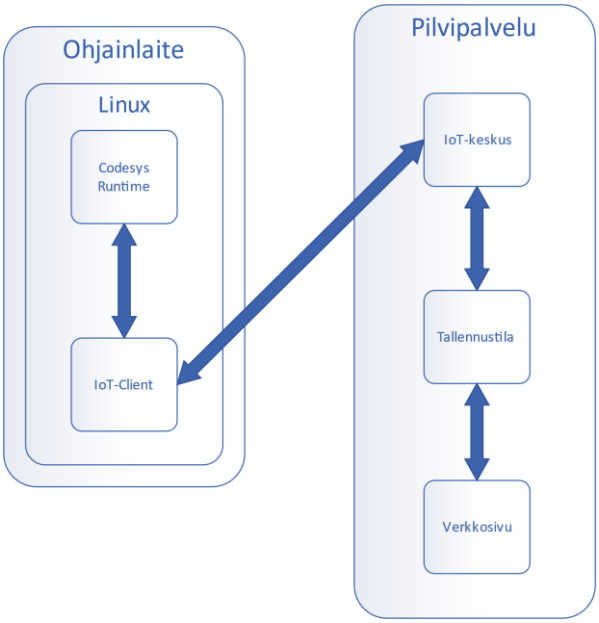
Työssä tutkitaan mahdollisuutta käyttää valmista palvelua visualisoinnin apuna. Tämä helpottaisi visualisointisivujen kokoonpanoa selkeään muotoon lyhyemmässä ajassa, koska verkkosivuilla näytettävät tiedot ovat konekohtaisia. Ajan säästö verrattuna tekstipohjaiseen ohjelmointitapaan, kuten JavaScript, olisi todennäköisesti merkittävä.

Palvelua kokeillaan prototyypissä, jos pilvipalveluntarjoajalta löytyy kyseinen palvelu. Visualisointipalvelun puuttuminen ei kuitenkaan ole ratkaiseva tekijä palveluntarjoajan valitsemiseen, koska oletuksena on, että palvelun tai sen visualisointien integrointi nettisi-

vulle ei onnistu aivan suoraan, jolloin tarvitaan kokonaan itse kehitetty ratkaisu. Itsetehtyyn ratkaisuun on valittu kieleksi JavaScript. JavaScript valittiin sen takia, että se on yleisesti käytetty ja sille löytyy useita kehitysympäristöjä ja kirjastoja visualisointiin. Sillä voi kehittää myös nettisivun toiminnan, jolloin ei tarvita ohjelmien välisiä rajapintoja.

### 3.6 Rakenteen yhteenveto

Dataa siirretään järjestelmän eri osien välillä. Ohjainlaitteen Codesys-suoritusympäristöstä saadaan dataa laitteen toimintaan liittyen, esimerkiksi lämpötiloja ja akun varaustila. Tiedot täytyy saada IoT-asiakassovelluksen käytettäväksi, jotta se voi kasata niistä oikean mallisen tietopaketin, joka lähetetään internetin välityksellä pilven IoT-keskukseen. IoT-keskus lähettää datan eteenpäin tallennustilaan paketissa, joka sisältää tarvittavat tiedot laitteesta, jotta data saadaan jaettua eri koneille kuuluvaksi. Verkkosivulta asiakas voi sitten valita nähtävät tiedot, jolloin palvelin hakee ne tallennustilasta ja muokkaa niistä halutun kaavion tai listan. Kuvassa 1 nähdään suunnitellun järjestelmän rakenne ja datan liike palveluiden ja sovellusten välillä.




Kuva 1. Järjestelmän rakenne ja datan liike

# 4 Palveluiden hinnat

Tässä luvussa verrataan Microsoft Azure-, Google Cloud-, Amazon Web Services -pilvipalveluiden hintoja edellisessä luvussa käsitellyille palveluille. Jokaiselle tarvittavalle palvelulle valitaan paras palveluntarjoaja. Kriteereinä ovat palvelun ominaisuudet, hinta ja käytön sujuvuus. Tärkeimpänä kriteerinä on se, että projektissa tarvittavat ominaisuudet löytyvät palvelusta. Jos ne löytyvät kaikilta palveluntarjoajilta, hinta otetaan huomioon. Huomioon on otettava myös toiminnan käytön ja kehittämisen ajallinen kesto, jonka voidaan olettaa olevan suuri osa järjestelmän kustannuksia. Jos hintaero palveluntarjoajien välillä ei ole merkittävä, voidaan palvelut ottaa samalta pilvipalvelulta tiedonsiirtomaksujen minimoimiseksi.

Jokainen tutkituista palveluntarjoajista hinnoittelee tuotteensa eri perusteilla, joten hintoja vertaillaan eri viestiko'oilla. Verkkosivuun ylläpitoon tarvittavan palvelimen tehoa ei ole projektissa arvioitu, mutta palvelinten hintojen oletetaan skaalautuvan lähes samalla tavalla jokaisella palveluntarjoajalla. Hinta-arviot tehdään palvelimilla, joissa on yksi virtuaaliydin ja 1 GB keskusmuistia, tai palvelimella, joka on lähinnä tätä teholuokkaa. Palvelimen täytyy myös olla toiminnassa jatkuvasti, joten ei voida käyttää halvempia palvelimia, jotka antavat laskentatehoa käytettäväksi riippuen palvelimien käyttöasteesta.

IoT-hallintapalvelun hinta lasketaan käyttäen oletuksia:

- Laitteita on 20 kappaletta.
- Tilaviestejä lähetetään minuutin välein.
- Viestikoko on 30 kB.
- Laitteet ovat yhteydessä 12 tuntia päivässä.

Lisäksi havainnollistetaan palvelujen skaalautuvuutta kaavioilla, joista näkee palvelujen hinnat viestikoon ollessa välillä 1...100 kB. Tallennustilaa laitteiden lähettämille tiedoille tarvitaan 158 GB yhtälön

$$S = N_m * N_l * S_v, \quad (1)$$

jossa S on tarvittava tallennustila (kB), $N_m$ on minuuttien määrä vuodessa ja $S_v$ on viestin koko (kB) mukaisesti. Muistin määrälle skaalautuvuus on tärkeä, koska todellinen datamäärä jää todennäköisesti pienemmäksi, ja ennen kuin kaikki kaksikymmentä laitetta ovat olleet toiminnassa vuoden, datamäärä vasta nousee kohti laskettua lukua.

## 4.1 Microsoft Azuren hinta

Microsoftin hinnat on laskettu käyttäen alueena Pohjois-Eurooppaa. Hinnat selvitettiin Azuren hintalaskimen avulla 24. marraskuuta 2020.

Microsoft Azuren IoT-hallintaan käytettävälle IoT Hub -palvelulle valittiin Standard-taso, koska Basic-tason palvelu ei sisällä mahdollisuutta tiedonsiirtoon pilvestä laitteelle.

Microsoftin IoT Hub -palvelun koko määräytyy viestien päivittäisen määrän perusteella. Lähetettävien ja vastaanotettavien viestien maksimikoko on 4 kB. Jos viestin koko ylittää maksimikoon, sen käyttämä viestimäärä saadaan jakamalla viestikoko maksimikoolla, eli viestin oletuskoolla 30 kB kuluu kahdeksan yksikköä per viesti. Laskuissa käytettävillä oletuksilla laskettaessa päivittäiseksi viestimääräksi saadaan 115 200 kappaletta. Tällöin toteutukseen riittää pienin Standard-tason palvelu, S1, johon kuuluu 400 000 viestiä. Palvelussa riittää laajennusvaraa, joten viestikokoa tai laitemäärää voidaan lisätä merkittävästi, ennen kuin tarvitsee perustaa toinen samanlainen palvelu. Palvelun hinta on 25 dollaria kuukaudessa.

Verkkosivuston ylläpitoon valittiin palvelinkoko B1s, jolla on yksi ydin ja 1 GB keskusmuistia. Tämä maksaa 8,25 $ kuukaudessa. B-sarjan (burstable) virtuaalikoneet ovat merkittävästi halvempia kuin muut saman tehoiset, mutta niitä voidaan käyttää suurella käyttöasteella vain osa ajasta. Ne toimivat alhaisemmalla käyttöasteella suuren osan ajasta, mikä kerää palvelimelle pisteitä. Kun käyttöaste menee tietyn rajan yli, palvelin sen sijaan kuluttaa pisteitä, ja pisteiden loppuessa liikenne rajoitetaan asetetulle rajalle. Valitulla palvelimella rajana on 10 %:n käyttöaste. Käyttöastetta valmiille ohjelmalle on vaikea ennustaa, mutta koska kyseessä on nettisivu, voidaan olettaa sen käytön olevan vaihtelevaa ja tällöin valitun tyypin palvelin olisi oikea ratkaisu.

Palvelimen lisäksi tarvitaan tallennustila laitteiden lähettämille tiedoille. Tähän vaihtoehtoina ovat hallittu ja hallitsematon muisti. Hallittu muisti eroaa hallitsemattomasta siten, että hallittua muisti ylläpitää palveluntarjoaja, ja se on valmiiksi kytkettynä virtuaalikoneeseen. Se on siis helpompi asentaa ja käyttää, eikä siihen pääse palvelimen ulkopuolelta käsiksi. Hallitsemattomassa muistissa kehittäjän täytyy itse konfiguroida yhteys palvelimen ja muistipalvelun tilin välille, ja siihen on mahdollisuus saada yhteys muualtakin kuin palvelimelta.

Parempi vaihtoehto projektiin on hallitsematon muisti, koska sen kustannukset ovat noin puolet hallitun muistin kustannuksista ja mahdollisuus ottaa yhteys muualtakin kuin palvelimelta antaa mahdollisuuden esimerkiksi tietojen lataamiseen suoraan tallennustilasta. Tätä ominaisuutta tarvitaan myös, jotta datan saa siirrettyä IoT Hub -palvelusta tallennustilaan.

Tietojen kirjoittaminen maksaa 6,5 $ per miljoona kirjoitusta, lukeminen 0,52 $ per miljoona kirjoitusta, joiden maksimikoko on 4 MB, ja tallennustila 0,022 $ gigatavua kohden. Oletetulle tarvitulle maksimitietomäärälle, 158 GB saadaan hinnaksi 5,98 dollaria, jos oletetaan, että jokainen viesti kirjoitetaan muistiin kerran, ja muistista luetaan tietoja noin kaksinkertainen määrä kuin sinne lähetetään. Toisena vaihtoehtona ollut hallittu muisti maksaisi 11,33 $, koska 128 GB jälkeen seuraava mahdollinen koko olisi 256 GB.

Palvelimelle vaihtoehtona ollut korkeamman tason palvelu on Azuressa nimellä App Service. Edullisin tällainen palvelu, johon kuuluu skaalautuvuus, olisi Standard-tason palvelu, jossa on 50 GB tallennustilaa, 1.75 GB keskusmuistia ja 1 ydin. Tämän palvelun hinta on 69,35 $. Vaikka muistia on enemmän, se on selvästi kalliimpi kuin B-sarjan virtuaalikoneet. Tähän siirtyminen on kuitenkin mahdollista, jos virtuaalikoneen ylläpito koetaan liian työlääksi ja jos tiedonsiirto IoT-keskuksesta tai tallennustilasta on mahdollista.

Taulukosta 1 nähdään, että kokonaishinnaksi saadaan halvemmalla vaihtoehdolla (vaihtoehto 1) 39,23 dollaria kuukaudessa, ja korkeamman tason palveluilla (vaihtoehto 2) 100,33 dollaria kuukaudessa.

Taulukko 1. Azure-pilvipalvelun osien hinnat

| Hinnat dollareina kuukaudessa, Azure | | |
|---|---|---|
| **Palvelu** | **Vaihtoehto 1** | **Vaihtoehto 2** |
| Iot Hub S1 | 25,00 | 25,00 |
| Virtuaalikone B1s | 8,25 | |
| Tallennustila 158 GB | 5,98 | 5,98 |
| App service Standard | | 69,35 |
| **Yhteensä** | **39,23** | **100,33** |

## 4.2 Amazon Web Services -hinta

AWS-hinnat on laskettu käyttäen palvelun alueena Sveitsin Frankfurtia ja samoja oletuksia viestimääristä ja koosta kuin aikaisemmin.

Amazon Web Servicen IoT Core -palvelun hinnoittelu poikkeaa merkittävästi Microsoftin mallista. Siinä palvelun hintaan vaikuttaa vain viestien sekä niistä aiheutuvien tapahtumien määrä. Amazonilla viesti jaetaan 5 kB:n osiin, jolla saadaan viestien kuukausittaiseksi kokonaismääräksi 2 592 000 kappaletta. Palvelu maksaa

- 0,096 $ per miljoona minuuttia yhteydessä
- 1,20 $ per miljoona viestiä, jos viestimäärä on alle miljardi
- 0,18 $ per miljoona tapahtumaa aktivoitu.

Tapahtumia ovat muun muassa tiedonsiirto IoT-hallintapalvelusta tallennustilaan, joten se tapahtuu aina, kun tietoja laite lähettää tietoja pilveen. Oletustiedoilla palvelun kuukausihinnaksi tulee 3,23 $ yhtälön 2 mukaisesti.

$$Kuukausihinta = 0{,}096\ \$ * \frac{t_{Cmin}}{10^6\ min} + 1{,}2\ \$ * \frac{\mathrm{N} * N_v}{10^6} + 0{,}18\ \$ * \frac{\mathrm{N}}{10^6} \quad (2)$$

$t_{Cmin}$ on aika yhdistettynä minuutteina

N on lähetettyjen viestien lukumäärä

$N_v$ on osien määrä, kun viesti jaetaan 5 kB osiin

Amazonin palvelu suosittelee, että jokaisella laitteella on yksilöllinen sertifikaatti palveluun yhdistämiseksi, jolloin IoT-ratkaisun sovelluksen asentaminen ja päivittäminen laitteille eivät onnistu samalla asennuspaketilla. Tämän takia IoT-palvelua ei oteta Amazonilta, vaikka hinta onkin paljon edullisempi kuin Azurella. Amazonin palvelussa on mahdollista käyttää samaa sertifikaattia testivaiheen helpottamiseksi, mutta se ei ole suositeltavaa tietoturvan kannalta.

Verkkosivupalvelimeksi valittiin samantapainen purskautettava palvelin kuin Azurelta, Sen malli on t3a.micro, jolla on kaksi ydintä ja yksi GiB keskusmuistia. Tämän kuukausihinnaksi saadaan 7,78 $, mutta kuten Azuren palvelimella tätä ei voida pitää luotettavana hintana mahdollisesti tarvittavan lisätehon ja redundanssin takia.

Tallennustilaksi on kaksi vaihtoehtoa kuten Azurella. Amazon S3 on halvempi vaihtoehto, joka ei ole liitettynä virtuaalikoneeseen. Sen hinta määräytyy seuraavien sääntöjen mukaan:

- 0,0245 $ per GB tallennettua dataa kuukaudessa
- 5,4 $ per miljoona kirjoitusta
- 0,43 $ per miljoona lukua.

Näiden avulla 158 GB muistille, jos jokainen viesti kerran kirjoitetaan ja luetaan kahdesti, saadaan hinnaksi 6,58 $. Virtuaalikoneeseen liitetyn muistin hinta määräytyy tallennustilan mukaan, ja sille saadaan hinnaksi 18,8 $. Tämän muistin palvelun nimi on Elastic Block Storage.

Taulukossa 2 nähdään Amazonin palveluiden kokonaishinnat halvemmalle ja kalliimmalle vaihtoehdolle. Halvempi maksaa 17,59 dollaria ja kalliimpi 29,81 dollaria. Vaihtoehtojen hintaero on pienempi kuin Azuressa, koska Amazonin palvelussa korkeamman tason ratkaisua vastaa Elastic Beanstalk -palvelu, joka auttaa käyttäjää valitsemaan oikeat palvelut, kuten palvelimen ja skaalautuminen, ja hinnat määräytyvät niiden käytön

mukaisesti. Laskuihin ei otettu huomioon lisäpalveluita, koska niiden käyttöä on tässä vaiheessa vaikea arvioida.

Taulukko 2. Amazon Web Service -palveluiden hinnat

| Hinnat dollareina kuukaudessa, AWS | | |
|---|---|---|
| **Palvelu** | **Vaihtoehto 1** | **Vaihtoehto 2** |
| IoT Core | 3,23 | 3,23 |
| Virtuaalikone T3a.micro | 7,78 | 7,78 |
| Tallennustila 158 GB | 6,58 | |
| Elastic Block Storage | | 18,8 |
| **Yhteensä** | **17,59** | **29,81** |

## 4.3 Google Cloudin hinta

Google Cloudin IoT Core -palvelun hinta määräytyy vain datamäärän perusteella. Ensimmäinen 250 MB on ilmaista, jonka jälkeen 250 GB asti hinta on 0,0045 $/MB. Sen jälkeen hinta halpenee. Järjestelmästä tehdyillä oletuksilla viestien kokonaisdatamääräksi kuukaudessa saadaan 12 960 MB, jolla kokonaishinnaksi saadaan 57,20 $. Hinta on merkittävästi suurempi kuin Microsoft Azuren IoT-palvelun, joka täyttää kaikki tarvittavat kriteerit, joten Google Cloudia ei valita projektiin. Googlen palvelu olisi pienillä datamäärillä halvempi kuin Azure, koska se skaalautuu täysin laitteiden määrän mukaan, mutta pahimmassa tapauksessa hinta olisi lähes kymmenkertainen Azureen verrattuna.

Google Cloudin IoT Core kävisi tietoturvan osalta projektin tarkoitukseen, sillä mikrokontrollereille tarkoitettu sertifikaatti on muuttumaton vuoteen 2030, joten jokaiselle laitteille ei tarvita yksilöllistä sertifikaattia.

Palvelimen valinnassa virtuaalikoneen hinnaksi saadaan 6,73 $, kun valitaan 2 ydintä ja 1 GB keskusmuistia sisältävä e2-micro-virtuaalikone. Sillä on sama toimintaperiaate kuin kilpailijoilta valituilla, eli palvelinta voidaan hetkellisesti kuormittaa voimakkaammin, mutta muuten käyttöasteen täytyy olla alhainen. Tallennustilan, 158 GB, hinnaksi saadaan 5,67 $, kun käytetään samoja oletuksia kuin aikaisemmin, joissa jokainen viesti kirjoitetaan muistiin kerran ja luetaan kaksi kertaa.

Taulukossa 3 nähdään Google Cloudin palveluiden hinnat. IoT-palvelun hinta on selvästi suurempi kuin toisilla pilvipalveluntarjoajilla, joten Google Cloud voidaan jättää vaihtoehdoista pois.

Taulukko 3. Google Cloud -palveluiden hinnat

| Hinnat dollareina kuukaudessa, Google Cloud | |
|---|---|
| **Palvelu** | |
| IoT Core | 57,20 |
| Virtuaalikone T3a.micro | 14,46 |
| Tallennustila 158 GB | 5,67 |
| **Yhteensä** | **77,33** |

## 4.4 Palvelun valinta

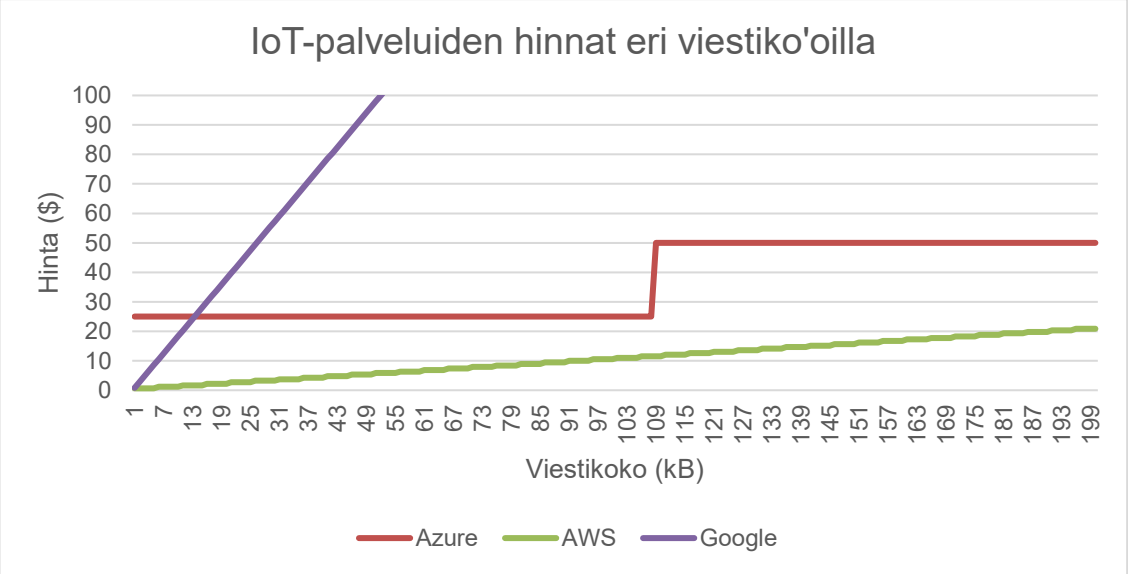
Virtuaalikoneiden ja tallennustilan hinnat ovat lähes samat Microsoftilla ja Amazonilla, ja vaikka Google Cloud on lähes kaksi kertaa kalliimpi, ei hintaero vielä tässä kokoluokassa ole merkittävä, kun verrataan kehityskustannuksiin tai IoT-palvelun hintaan. Suurin merkitys valinnassa on siis IoT-palveluiden hinnoilla. Taulukossa 4 nähdään eri palveluntarjoajien halvimmat hinnat vertailtuna.

Taulukko 4. Palveluntarjoajien hinnat

| Hinnat dollareina kuukaudessa | | | |
|---|---|---|---|
| **Palveluntarjoaja** | **Azure** | **AWS** | **Google Cloud** |
| Iot-keskus | 25,00 | 3,23 | 57,20 |
| Virtuaalikone | 8,25 | 7,78 | 14,46 |
| Tallennustila 158 GB | 5,98 | 6,58 | 5,67 |
| **Yhteensä** | **39,23** | **17,59** | **77,33** |

Amazonin IoT-palvelu on selvästi halvin tutkituista, mutta sen ominaisuudet eivät sovellu projektin tarpeisiin, yksilöllisten sertifikaattien tarpeen takia. Google Cloud taas on selvästi kallein IoT:n ja virtuaalikoneen osalta.

Kuvasta 2 nähdään IoT-palveluiden skaalautuvuus kahdellakymmenellä laitteella viestikoon ollessa välillä 1…100 kB. Samaa kuvaa voidaan käyttää laitemäärän muuttuessa, sillä laitemäärän vaikutus on suhteessa lähes sama kuin viestikoon. Jos laitemäärä kaksinkertaistuu, on sen vaikutus lähes sama kuin viestikoon kaksinkertaistuessa, jos viestikoko on reilusti yli viestin maksimikoon (4 kB Azurella ja 5 kB Amazonilla).




Kuva 2. IoT-palveluiden hinnat eri viestiko'oilla

Kuten kuvasta nähdään, Google Cloud on pienellä datamäärällä halvempi kuin Azure, mutta suuremmilla nousee todella korkeaksi verrattuna kahteen muuhun palveluun. Amazonin hinta skaalautuu yhtä lineaarisesti kuin Google Cloudin, mutta on paljon edullisempi.

Google Cloudin hinnan ja AWS:n IoT:n yksilöllisen sertifikaattitarpeen takia projektiin sopii parhaiten Microsoftin Azure IoT -palvelut, ja virtuaalikoneen hintojen ollessa lähellä toisiaan myös verkkosivun kehitykseen tarvittavat palvelut sijoitetaan Microsoftin pilvipalveluun ylimääräisten tiedonsiirtomaksujen välttämiseksi.

### 4.5 Palvelu visualisoinnin helpottamiseksi

Projektia varten yritettiin löytää pilvipalveluntarjoajilta datan esittämistä helpottavia palveluita. Palvelua etsittiin ensisijaisesti Azuren palvelusta, koska järjestelmän muut pilvipalvelut päätettiin sijoittaa sinne. Visualisointiin soveltuvat palvelut olivat pääosin liiketoimintatiedon hallintajärjestelmiä, joiden visualisoinnit näyttivät sopivilta, mutta epäselväksi jäi, ovatko palvelut oikeasti sopivia. Raporttien julkaiseminen nettisivulla onnistuu, mutta selvitettäväksi jäi, onnistuuko kaksisuuntainen dataliikenne helposti tällä tavalla.

Epäselvää on myös, helpottaisiko palvelu merkittävästi kehitystyötä. Toisena vaihtoehtona pidetty JavaScript-kirjaston käyttö visualisoinnin luomiseen on varmasti aluksi työläämpi, mutta kun tarvittavien komponenttien, kuten taulukoiden ja mittareiden, pohjat on luotu, kehitystyö helpottuu. Kummassakin tapauksessa kehittäjältä vaaditaan taitoja nettisivun luomiseen ja ohjelmointiin, joka tämän työn puitteissa jäi vajavaiseksi.

## 5 Pilviosuuden toteutus

Luvussa käydään läpi pilvipalvelun konfigurointi ja siihen liittyvien sovellusten asennus ja käyttö. Tarkoituksena on saada yhteys pilvipalveluun ja data liikkumaan laitteelta palvelimelle.

### 5.1 Ohjainlaitteen ohjelmisto

Ohjainlaitteelle tarvitaan sovellus tai sovelluksia, jotka hoitavat tietoliikenteen pilvipalvelun IoT-keskuksen ja ohjainlaitteen välillä.

Osuuden ensimmäisenä tavoitteena oli saada toimiva yhteys pilvipalveluun. Yhteyden muodostamista yritettiin ensin Raspberry Pi 4 -tietokoneella ja Rasbian-käyttöjärjestelmällä, jotta työskentely varsinaisen ohjainlaitteen riisutummalla Linuxilla olisi kokemuksen myötä helpompaa. Yhteyden muodostus ja tietojen lähetys onnistui Azuren Node- ja Python-ohjelmistokehityspakettien avulla. Projektissa oli kuitenkin tarkoituksena käyttää C-kieltä, jonka käyttö ei onnistunut, mikä johtui ohjelmistokehityspaketin käyttämien kirjastojen puutteesta.

Projektin varsinaisen EPEC 6107 -ohjainlaitteen erittäin riisuttu Linux-käyttöjärjestelmä ei sisällä paketinhallintajärjestelmää, joka helpottaisi tarvittavien sovellusten asennusta ja hallintaa. Paketinhallintajärjestelmänkään asennus ei onnistunut paketin asennusohjeissa käytetyn ohjelman puutteen takia. Projektin osuus koskien ohjainlaitteen konfigurointia ja ohjelmointia jätettiin teorian tasolle, koska siihen olisi tarvittu odotettua enemmän osaamista Linuxin puolelta.

Sovelluksen tehtäviä ovat

- yhteyden muodostus pilvipalveluun
- tietojen kerääminen muistiin
- datan muotoilu haluttuun lähetysmuotoon
- tietojen lähetys pilveen
- tiedostojen ja parametrien vastaanottaminen pilvestä
- parametrien välitys niitä käyttävään ohjelmaan.

Yhteyden muodostuksessa olisi käytetty ohjainlaitteen sarjanumeroa laitteiden identifioinnissa, jolloin kaikkiin laitteisiin olisi voitu asentaa sama sovellus ilman ylimääräisiä yksilöllisiä tunnuksia, ja laitteet olisi silti voitu erottaa toisistaan. Azuren nettipalvelussa on mahdollista asettaa avain ja laitetunnus, joten siellä olisi luotu jokaiselle ohjainlaitteelle tunnukset niiden sarjanumeroiden mukaan. Näin laitteet olisivat nettiyhteyden löytäessään saaneet muodostettua yhteyden.

Lähetettävät datat saadaan ohjainlaitteella suoritettavasta Codesys-suoritusympäristöstä. Dataa täytyy tallentaa lähetysten väliseltä ajalta, mihin käytetään ohjainlaitteen keskusmuistia. Datan väliaikainen säilytys olisi mahdollista kirjoittamalla datat tekstitiedostoon, mutta silloin muistin elinkaarta kuormitetaan kirjoittamalla ja pyyhkimällä sitä jatkuvasti. Ohjainlaitteella on 1024 MB keskusmuistia, joten sen ei pitäisi olla rajoitteena, koska viestikoko on sadan kilotavun luokkaa.

Kerätyt datat lähetetään JSON-muodossa pilveen, jolloin niiden käyttö on JavaScriptillä helppoa. JSON (JavaScript Object Notation) on yksinkertainen tiedostomuoto tiedonvälitykseen, jossa tietoja ryhmitellään tekstimuodossa nimi- ja arvo pareihin. Arvona voi olla myös uusia nimi-arvo-pareja. Työssä datan lisäksi täytyy lähettää datan aikaleima sekä nimi. Työssä käytettävä asettelu voisi olla esimerkkikoodissa 1 näkyvän mukainen.

```
{
    "name": "motor rpm",
    "values": [
        {
            "time":"2020-05-12T11:03:25",
            "value": "1700"
        },
        {
            "time":"2020-05-12T11:03:26",
            "value": "1750"
        },
        {
            "time":"2020-05-12T11:03:27",
            "value": "1740"
        },
        {
            "time":"2020-05-12T11:03:28",
            "value": "1800"
        }
    ]
}
```

Esimerkkikoodi 1. Lähetettävän datan muoto

Datan Lähetykseen ja vastaanottamiseen kannattaa käyttää Microsoftin Azure-kehityspakettien työkaluja. Datan tallennukseen ja muuntamiseen JSON-muotoon tarvitaan ulkopuolinen tai itse kehitetty sovellus. Myös vastaanotettujen tiedostojen ja parametrien käyttöön tai välittämiseen täytyy kehittää oma toteutus. Parametrit täytyy välittää Codesys-kehitysympäristöön, ja tiedostot voidaan käsitellä eri tavoilla riippuen niiden tyypistä. Päivitystiedostot voidaan purkaa ja asettaa oikeaan kansioon ja konfiguraatiotiedostot niiden omiin kansioihinsa. Näiden asioiden toteuttamiseksi tarvitaan paljon osaamista Linux-ympäristöihin ja ohjelmointiin liittyen.

## 5.2 Yleistä Azuren palveluiden rakenteesta

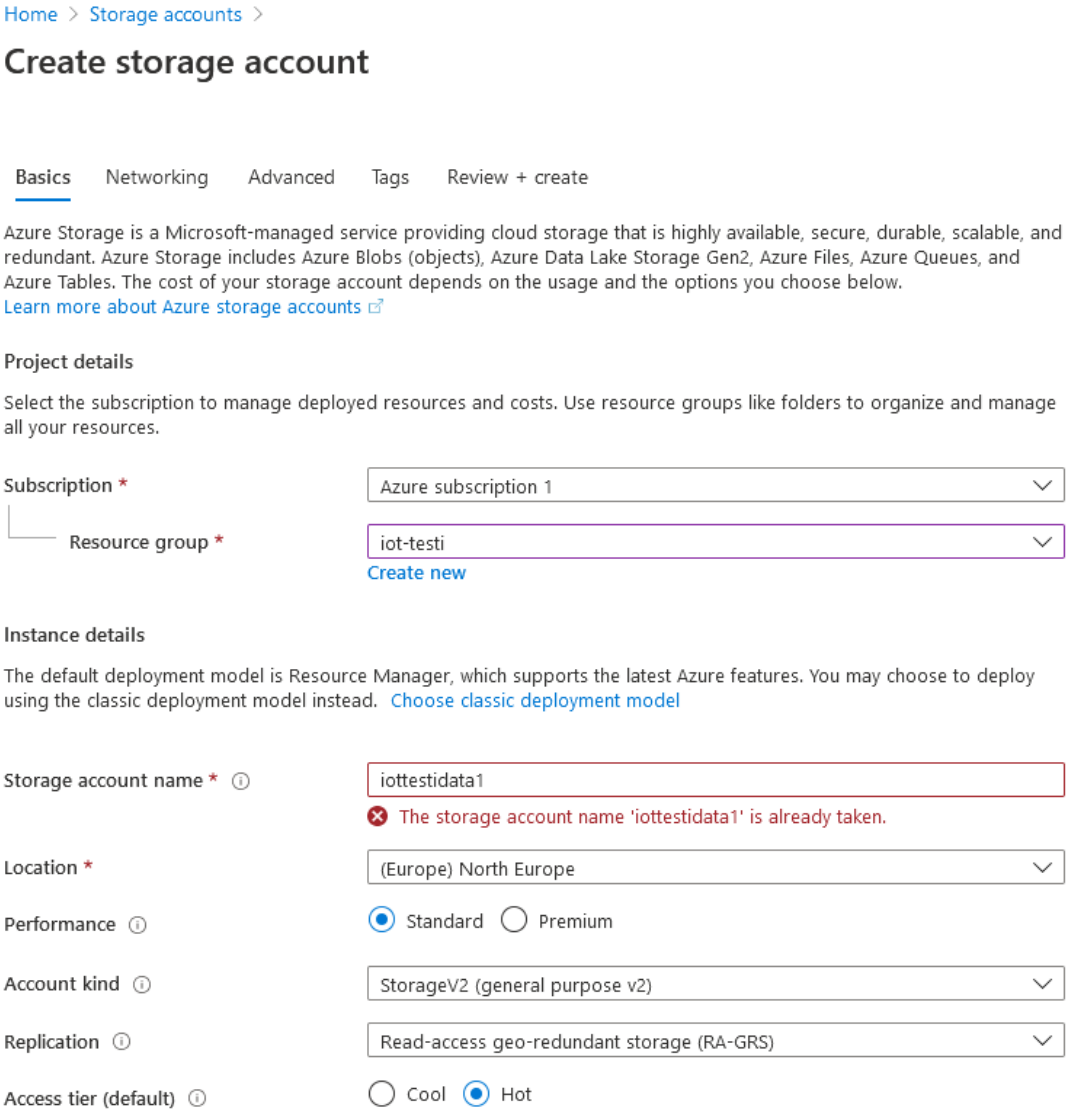
Microsoft Azuren pilvipalvelun rakenteeseen ja käyttöön liittyy muutama yleinen asia, jotka on hyvä tietää. Niitä käydään läpi tässä luvussa.

Palvelun käyttämiseksi tarvitaan tili. Tileille voidaan antaa eri rooleja, joiden mukaan tilien oikeudet muuttuvat. Mahdollisia rooleja ovat esimerkiksi tilien avustaja (contributor), lukija ja laskutietojen lukija. Yrityksellä on siis mahdollisuus hallita useita tilejä ja antaa niille oikeuksia eri palveluihin ja estää niiden pääsy toisiin. Tämä mahdollistaa paremman hallittavuuden käyttöoikeuksille, koska eri henkilöt saavat oikeudet vain tarvitsemiinsa toimintoihin ja resursseihin.

Eri resursseja, eli palveluita, varten perustetaan resurssiryhmiä (Resource Group), joiden avulla jaotellaan luotuja palveluita ryhmiin esimerkiksi projektien mukaan. Jaottelun avulla saadaan resurssiryhmäkohtaiset kustannustiedot ja nähdään kaikki projektin käyttämät palvelut yhdestä paikasta. Sen avulla voidaan myös tarvittaessa helpommin kopioida kokonaisuuksia esimerkiksi redundanssin tai skaalautuvuuden helpottamiseksi. Tähän projektiin luotiin resurssiryhmä iot-testi.

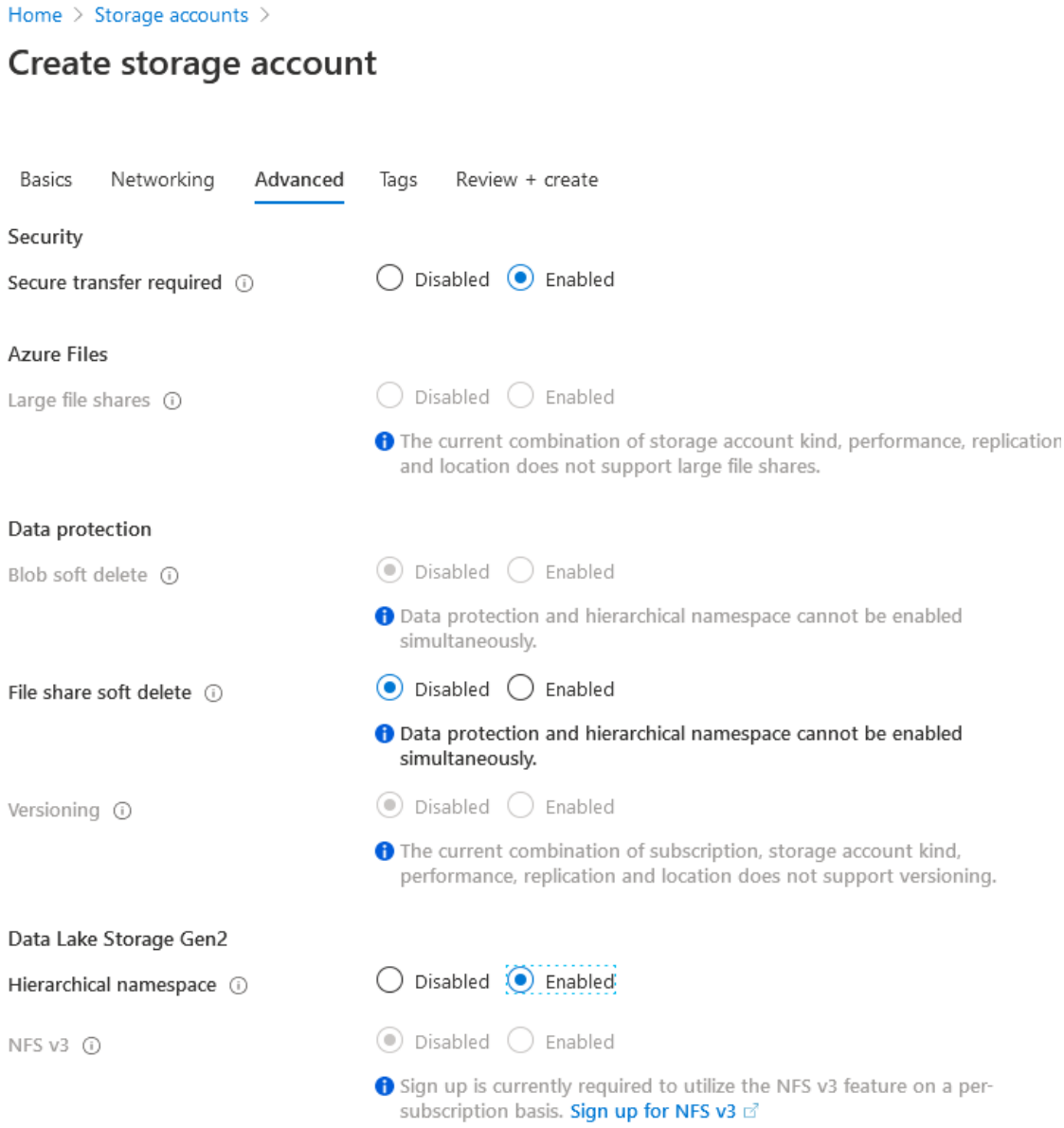
## 5.3 Datan tallennustila

Projektiin täytyi luoda tili Azuren Storage -palveluun IoT-laitteiden lähettämää dataa varten. Tilille tarvitaan nimi, sijainti, tilin tyyppi ja datan redundanssin asetukset. Tiliä luodessa valitaan sen käyttämä resurssiryhmä, eli tässä tapauksessa projektia varten luotu iot-testi. Sijainniksi valittiin Pohjois-Eurooppa ja tilin tyypiksi StorageV2, joka on uusin tilityyppi ja tukee kaikkia palvelun ominaisuuksia, joita muutkin mahdollisuudet tukevat. Datan redundanssiasetukseksi valitaan LRS (locally redundant storage), joka on paikallisesti redundanssi, eli data on kopioitu kolmesti samaan fyysiseen sijaintiin. Se ei auta, jos sen datakeskus menettää kaiken datansa, mutta testikäyttöön se on hyvä edullisuutensa takia. Muita vaihtoehtoja ovat alue- ja mantereredundantit tallennukset, jotka suojaavat paremmin yksittäisen datakeskuksen ongelmilta. Nämä asetusten valinnat nähdään kuvassa 3.




Kuva 3. Tallennustilin luonti: Basics-välilehti

Lisäksi Advanced-välilehdeltä täytyi sallia ”Hierarchical namespace”, jotta IoT-keskus voi välittää datan tallennustilaan. Ilman tätä asetusta IoT-keskuksen siirtämät datat eivät pääse tallennustilaan asti. Kuvassa 4 nähdään tämä asetus sekä muut asetukset toiselta välilehdeltä.

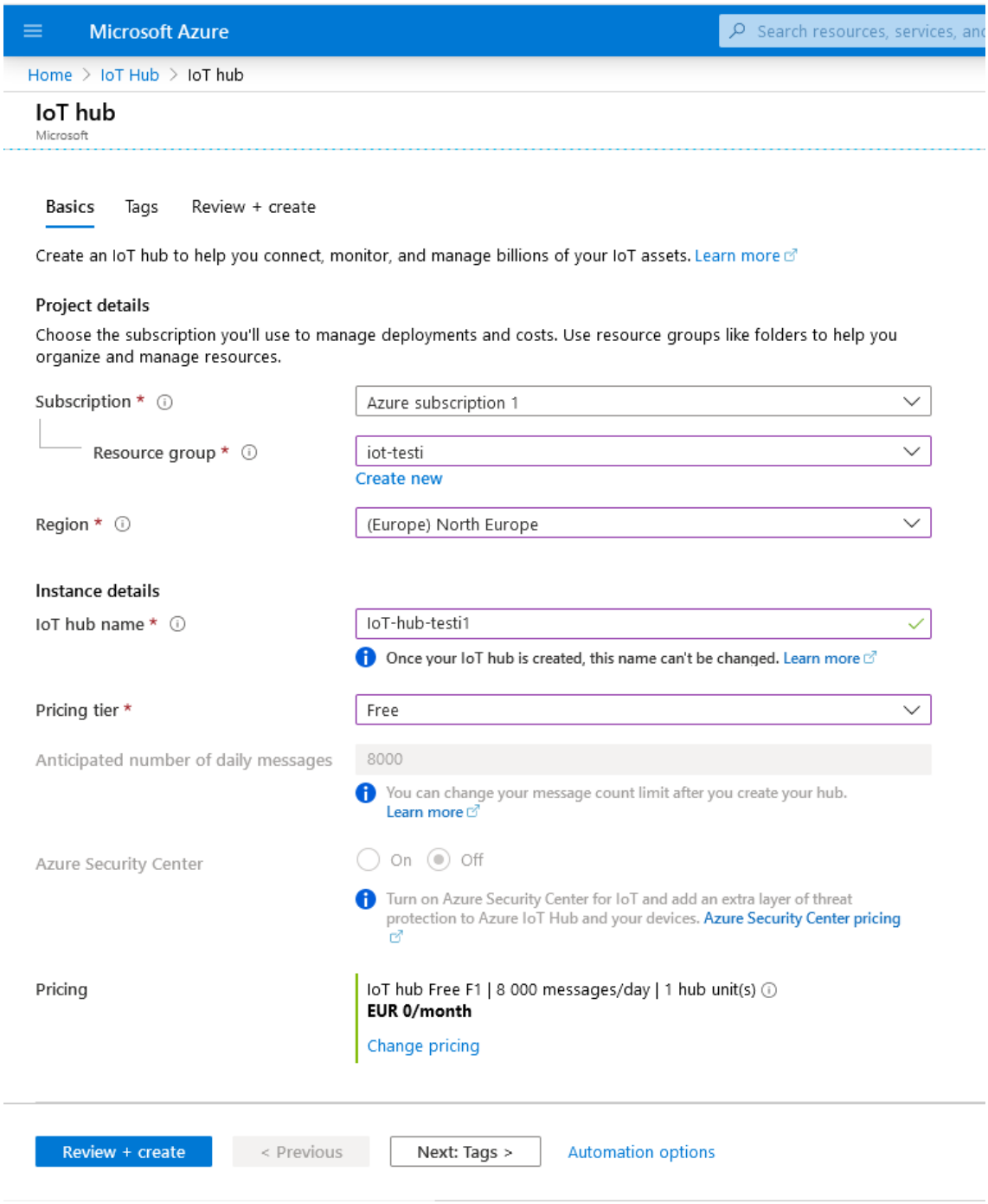



Kuva 4. Tallennustilin luonti: Advanced-välilehti

## 5.4 IoT-keskuksen luominen

IoT-keskuksen nimeksi päätettiin IoT-hub-testi1 ja se liitettiin resurssiryhmään iot-testi. Alueeksi valittiin Pohjois-Eurooppa ja laskutustasoksi Free. Varsinaiseen toteutukseen valittaisiin tason Standard-palvelu, mutta testausvaiheeseen riittää ilmainen versio, koska ensimmäisenä tavoitteena on vain saada viestit näkymään tallennustilassa eikä

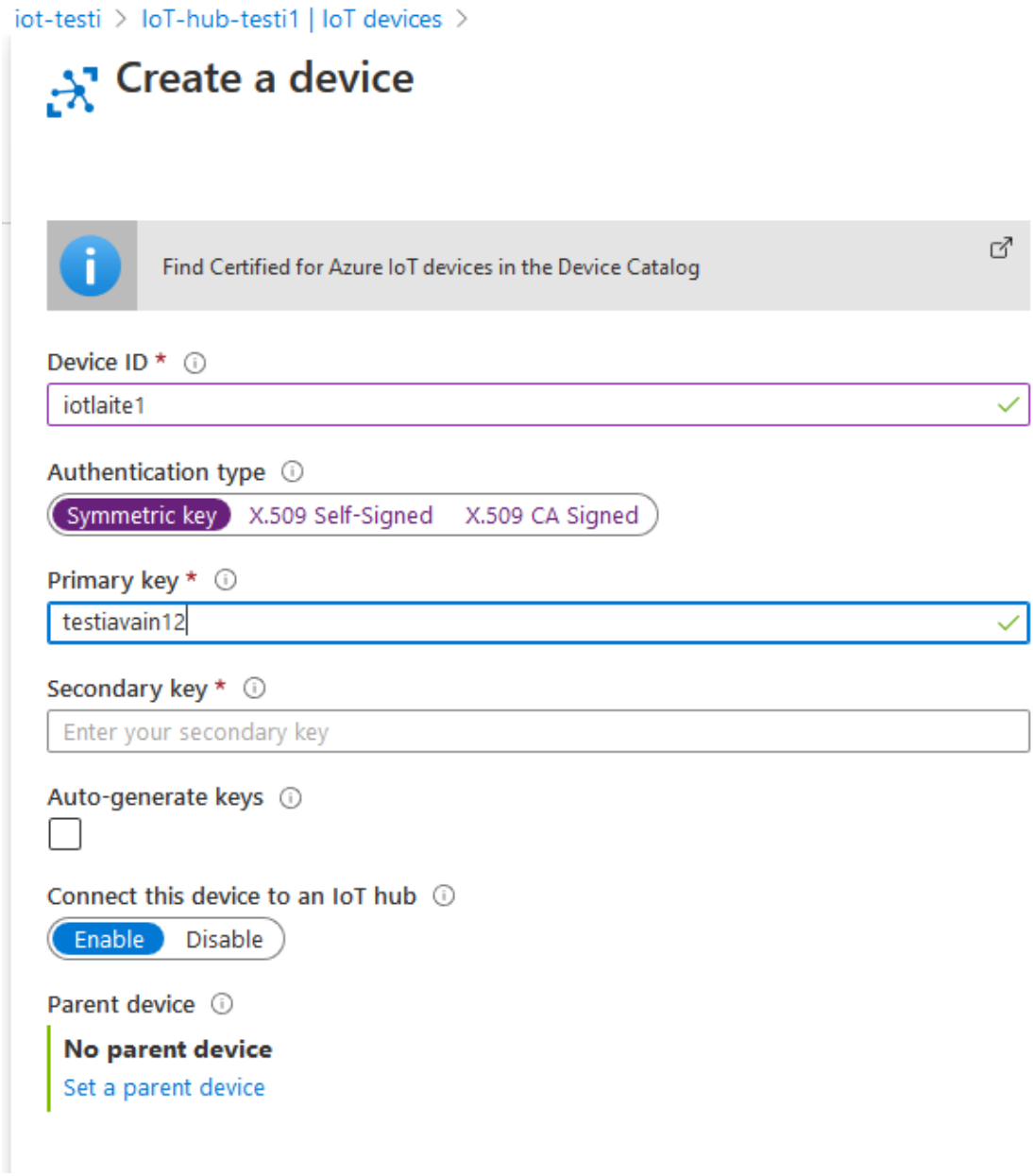
toteuttaa järjestelmän muita vaatimuksia. IoT-keskuksen luonnin asetukset nähdään kuvasta 5.




Kuva 5. IoT-keskuksen luonti

IoT-keskukseen täytyy lisätä myös laitteet, jotka siihen kuuluvat. Se tehdään IoT-keskuksen asetuksissa. Laitteelle annetaan tunniste sekä avain, joiden avulla laitteet saavat

yhteyden keskukseen. Testilaitteelle annettiin nimeksi ”iotlaite1” ja avaimeksi ”testiavain12”. Kuvassa 6 nähdään vaihtoehdot IoT-laitteen luontiin.




Kuva 6. IoT-laitteen luonti

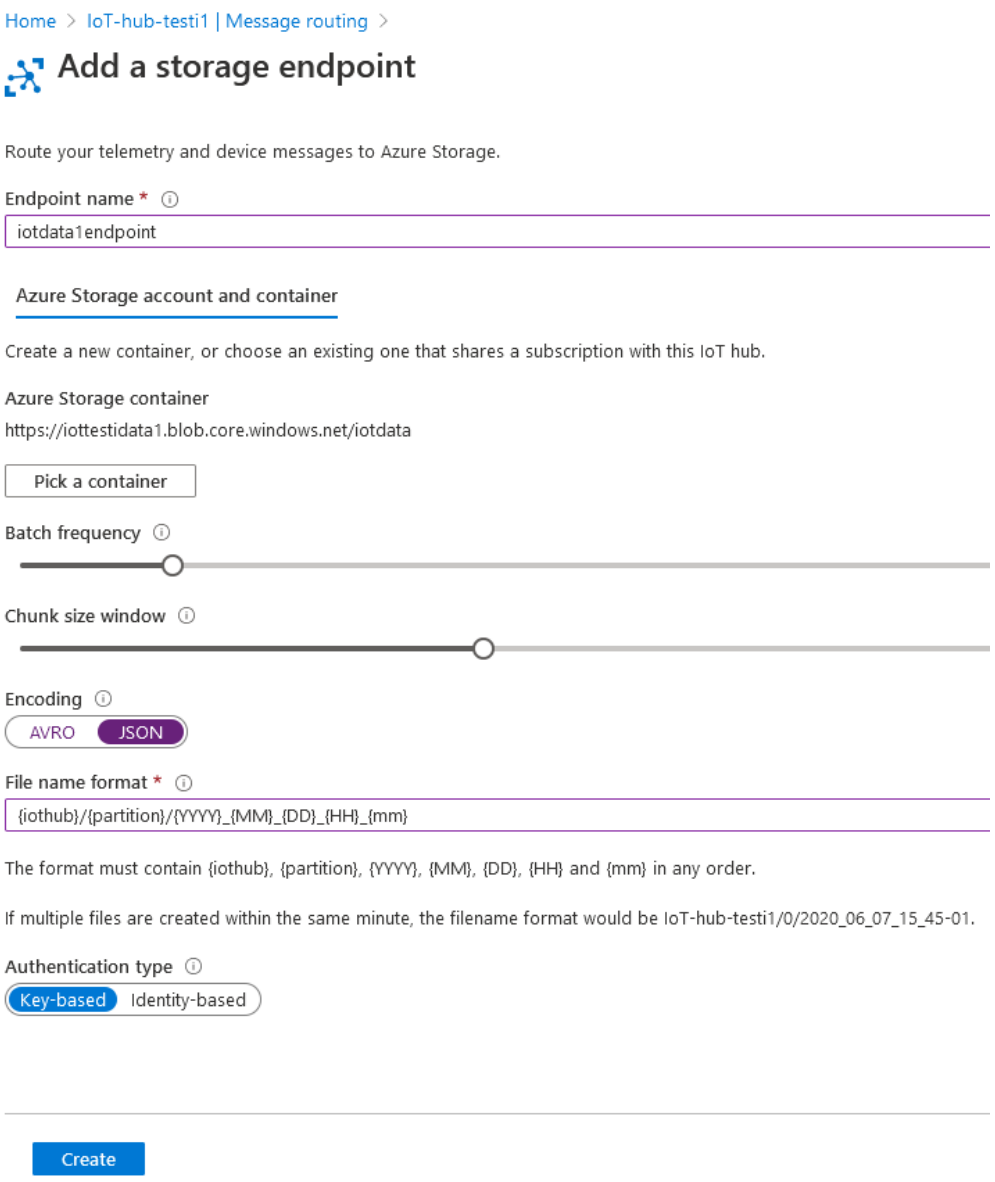
Nyt IoT-keskuksella on valmius ottaa laitteelta viestejä vastaan, mutta dataa ei vielä välitetä mihinkään. Siihen tarvitaan viestin välitystä (Message routing). Sen avulla viestit lähetetään IoT-keskuksesta eteenpäin, haluttuun palveluun. Viestejä voidaan ohjata tallennustilan lisäksi esimerkiksi Event Hub -palveluun. Se on palvelu, joka jakaa dataa

muualle ja josta muut palvelut voivat lukea dataa. Se toimii siis rajapintana IoT-keskuksen ja datan käyttäjien välillä. Palvelua ei kuitenkaan projektissa tarvita, koska datan ohjaaminen suoraan tallennustilaan onnistuu myös. Jos data ohjattaisiin toisen palveluntarjoajan tallennustilaan, täytyisi mahdollisesti käyttää Event Hub -palvelua.

Viestien ohjaamiseksi tallennustilaan luodaan uusi päätepiste valitsemalla ”Custom Endpoint” ja ”storage”. Luontisivulla valitaan tallennustilillä oleva ”astia” (container), johon tiedot tallennetaan. Astioiden avulla dataa voidaan jaotella tallennustilin sisällä. Tässä tapauksessa valitaan juuri luotu tili ”iottestidata1” ja luodaan sinne astia ”iotdata”. Lisäksi valitaan datalle JSON-muoto.

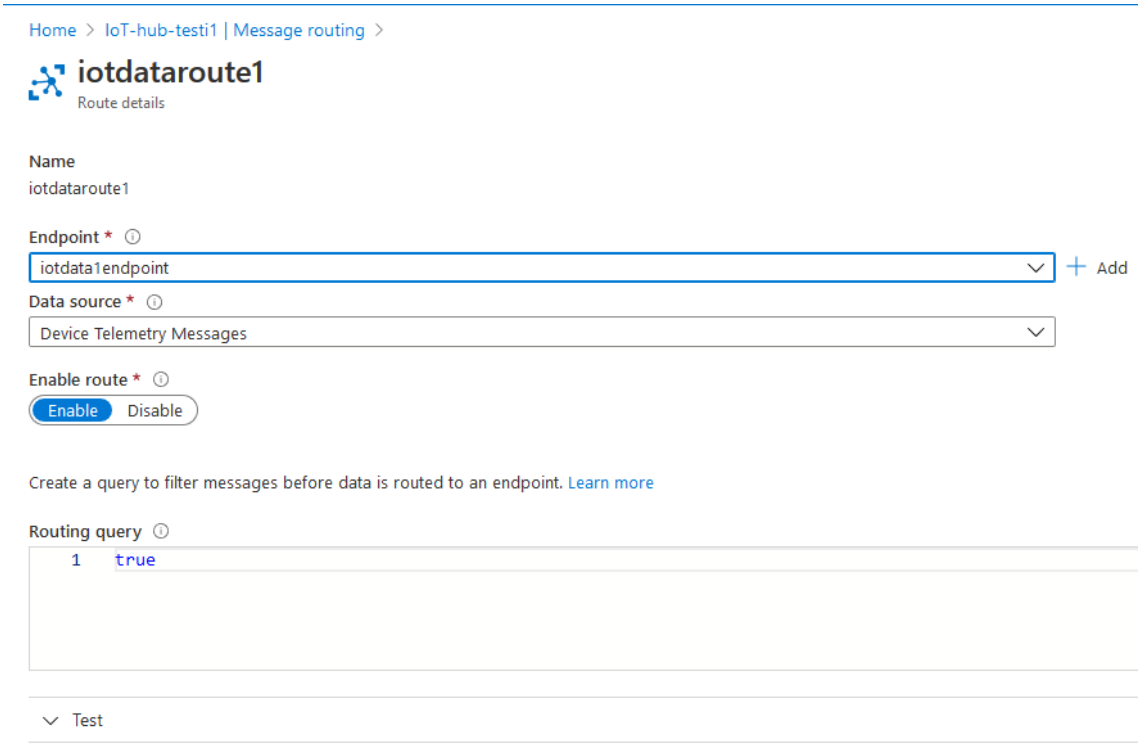
JSON (JavaScript Object Notation) on tiedostonvälitykseen tarkoitettu tiedostomuoto, jossa datalle annetaan atribuutteja ja näille arvoja. JSON-data tallennetaan tekstimuodossa, joten se on ymmärrettävässä muodossa ihmiselle. Avro on Apachen kehittämä JSON:in kaltainen tiedostomuoto, mutta se voidaan tallentaa ja siirtää myös binäärimuodossa. Sillä on myös muita tilaa säästäviä ominaisuuksia. Lopulliseen toteutukseen Avro olisi optimaalisempi vaihtoehto, mutta testiä helpottaa, että data on tekstimuotoista.

Viimeiseksi valitaan haluttu tiedoston nimeämismuoto. Nimeämismuotoilulla voidaan valita kansiorakenne lisäämällä tai poistamalla vinoviivoja. Asetukset näkyvät kuvassa 7.




Kuva 7. Päätepisteen luonti

Varsinainen reitti datalle luodaan valitsemalla luotu päätepiste. Reitille voidaan antaa myös koodimuotoisia ehtoja datan välittämiselle, mutta nyt oletusasetukset riittivät. Oletuksena reitti välittää laitteiden lähettämät telemetriaviestit, joita myös tässä testissä käytetään. Mahdollista on myös datan muokkaaminen tai poimiminen tässä vaiheessa, jolloin merkitsevä data saadaan poimittua. Tähän käytetään ”Routing query” -kohtaa, jossa voidaan käyttää JavaScript ja SQL -kieliä. Kuvassa 8 nähdään valitut asetukset reitille.




Kuva 8. Reitin luonti

## 5.5 Virtuaalikone

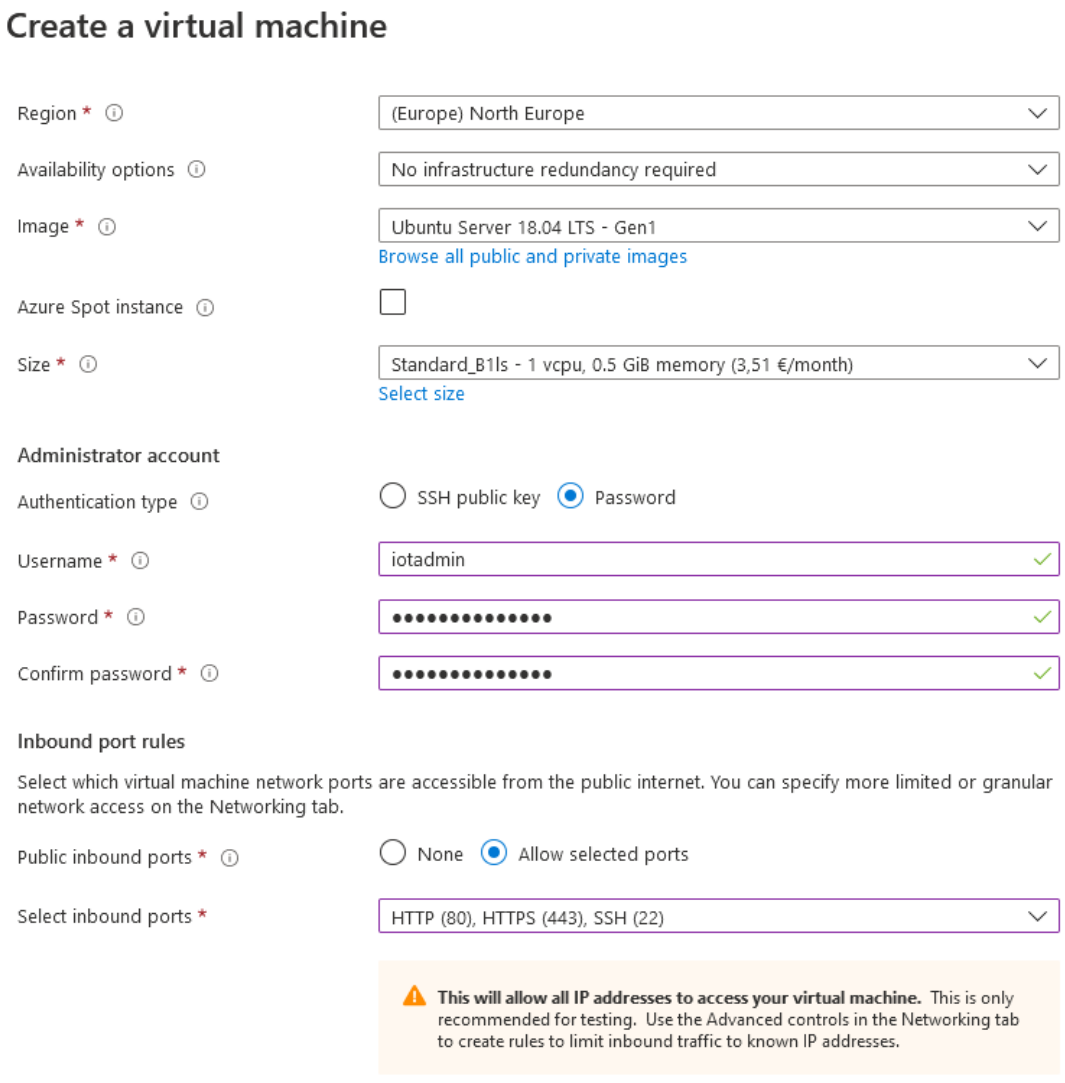
Virtuaalikonetta käytetään nettisivun palvelimena. Siellä tapahtuu kaikki varsinainen tiedon käsittely ja esitys asiakkaalle.

Virtuaalikoneen luonnissa valitaan nimen ja resurssiryhmän lisäksi

- käyttöjärjestelmä
- virtuaalikoneen koko
- massamuistin ominaisuudet
- ylläpitotilin asetukset
- sallittava liikenne

- tietoverkko, johon kone yhdistetään.

Käyttöjärjestelmäksi valittiin Ubuntu Server ja virtuaalikoneen kooksi pienin mahdollinen, jolla on yksi virtuaaliydin ja 0,5 GiB keskusmuistia. Koneen kokoa voi helposti kasvattaa, jos teho loppuu kesken, mutta testit tehtiin halvimmalla mahdollisella. Virtuaalikoneeseen otetaan yhteys SSH:lla, joten virtuaalikoneelle annetaan ylläpitäjän tunnus ja salasana. Lisäksi sallitaan SSH-, HTTP- ja HTTPS-liikenteet. Kuvassa 9 nähdään valitut perusasetukset.




Kuva 9. Virtuaalikoneen perusasetukset

Disks-välilehdellä valitaan virtuaalikoneelle massamuistin ominaisuudet. Tallennustilan tyypin voi valita kolmesta eri vaihtoehdosta; Premium SSD on nopein ja kallein mahdollinen, ja mahdollistaa suurimman tietoliikennemäärän. Standard SSD on halvempi ja hitaampi vaihtoehto tälle ja Standard HDD on kiintolevytila, joka mahdollistaa tietoliikennemäärän, joka on SSD-levyjen väliltä, mutta luku ja kirjoitusnopeudet ovat pienemmät. Hinnaltaan kiintolevy on halvin. Toteutukseen valittiin Standard SSD, koska tietoliikennemäärät eivät ole suuria.

Virtuaalikoneessa käytetään Azuren hallitsemaa muistia, jolle ei tarvita omaa säiliötä ja tallennustiliä, mikä helpottaa useiden virtuaalikoneiden luomista samalla mallilla. Virtuaalikoneen tarvitsema levytila ei myöskään ole suuri, joten levyn hinta ei ole merkittävä. IoT-dataa ei pysty lähettämään suoraan virtuaalikoneen levylle, vaan siihen tarvitaan tallennustili joka tapauksessa, ja suurin merkitys hintaan on datan säilytykseen käytettävän massamuistin tyypillä.

Networking-välilehdellä valitaan verkko, johon virtuaalikone liitetään. Tässä tapauksessa se liitetään oletuksena valittavaan projektin resurssiryhmän virtuaaliseen verkkoon ”iot-testi-vnet. Tarvittaessa tämä mahdollistaisi yhteyden usean virtuaalikoneen välille.

## 5.6 Virtuaalikoneen ja tallennustilan yhdistäminen

Koska järjestelmään liitettävät laitteet lähettävät dataa IoT-keskuksen kautta tallennustilaan, joka on Azuren Storage -palvelussa, täytyy virtuaalikoneen ja tallennustilan välille luoda yhteys. Tallennettu data on erillisellä palvelimella, josta tiedot täytyy ladata. Microsoft on kehittänyt työkalun, joka yhdistää tallennuspalvelun säiliön virtuaalikoneeseen lähes kuin normaalina kiintolevynä, jolloin tiedostoja voidaan hakea Linuxin tiedostojärjestelmän kautta. Työkalun nimi on Blobfuse.

Työkalu asennettiin Microsoftin ohjeiden mukaan. Ensin työkalu ladattiin ja asennettiin, jonka jälkeen virtuaalikoneesta valittiin välimuisti, johon avatut tiedostot ladataan. Välimuistin tarkoituksena on nopeampana muistina antaa ohjelmalle parempi suorituskyky, jotta datan lukeminen olisi lähempänä oikeaa kiintolevyä. Välimuistina käytettiin virtuaa-

likoneen väliaikaista tallennustilaa, jota käytetään käynnistykseen. Toisena mahdollisuutena olisi käyttää välimuistina keskusmuistia, mutta valitussa virtuaalikoneessa keskusmuistin määrä ei riitä siihen.

Ennen ohjelman käynnistämistä tarvitaan tiedosto, jossa on tallennustilin nimi, salasana tai yhdistysavain ja säiliön nimi. Seuraavaksi luodaan kansio, johon virtuaalinen kansio muodostetaan, jonka jälkeen blobfuse-sovellus käynnistetään esimerkkikoodin 2 mukaisella komennolla. Ohjeessa olevaan komentoon lisättiin allow_other-komento, joka antaa käyttöoikeuden kansioon.

```
sudo blobfuse ~/mycontainer --tmp-path=/mnt/blobfusetmp  --config-
file=/home/iotadmin/fuse_connection.cfg -o attr_timeout=240 -o en-
try_timeout=240 -o negative_timeout=120 -o allow_other
```

Esimerkkikoodi 2. Blobfuse-sovelluksen käynnistys

Tämän jälkeen kansiosta ”/mnt/blobfusetmp/” voidaan lukea IoT-laitteiden lähettämiä tiedostoja, jos tiedetään niiden nimet. Normaalista kansiosta poiketen tiedostojen ja kansioiden nimien täyttö tabulaattoria käyttäen komentorivillä ei toiminut, vaan yhteys katkesi, jonka jälkeen se piti katkaista ja aloittaa uudestaan. Tämä ei kuitenkaan ole ongelma, koska koko polulla tiedostoja avattaessa toteutus toimii, ja jos yritetään avata tiedostoa, jota ei ole olemassa, ohjelma ilmoittaa, että tiedostoa ei ole olemassa.

# 6 Pilvipalvelun tietoturva

Tässä luvussa perehdytään pilvipalvelun tietoturvaan ja verrataan sen riskejä paikallisen palvelimen riskeihin. Vertailussa oletetaan pilvipalvelun olevan suuri toimija, joka on kovan kilpailun takia ottanut turvallisuuden hyvin vakavasti. Pilvipalvelua valittaessa jää ostajan vastuulle selvittää palveluntarjoajan turvallisuus. Lopuksi pohditaan myös, mitä vaikutuksia ja vaaroja mahdollinen tietomurto voi aiheuttaa järjestelmässä.

## 6.1 Fyysinen tietoturva

Datakeskuksen ylläpitäminen on kallista. Sille tarvitaan hyvä sijainti luotettavien tietoliikenne- ja sähköyhteyksien takia. Keskuksen täytyy myös kestää paikalliset luonnonmullistukset sekä murtautumisyritykset. Lisäksi tarvitaan henkilökuntaa ylläpitämään keskusta sekä sen myötä henkilönvalvontaan liittyviä asioita, kuten kulunvalvontaa ja valvontakameroita, jotta mahdolliset pahat toimijat saadaan kiinni tai estettyä. (2, s. 26.)

Pilvipalveluntarjoajat ovat jo tehneet tarvittavat investoinnit datakeskuksen rakentamiseen ja ylläpitoon liittyen, ja niiden kustannukset jaetaan kaikkien käyttäjien kesken (2, s. 26). Tämän takia kustannusten määrä jää todennäköisesti pienemmäksi tilattaessa palvelin pilvipalveluntarjoajalta kuin rakentamalla se itse, ja ainakaan ei tule suuria alkuinvestointeja. Pienessä projektissa suurta datakeskusta ei tarvita, mutta jos palvelinta ylläpidetään toimistorakennuksessa, eikä ympäristön vaaroihin tai kulunvalvontaan ole kiinnitetty huomiota, on pilvipalvelun käyttö lähes varmasti tietoturvallisempi vaihtoehto.

## 6.2 Muu tietoturva

Tietoturvaan kuuluu myös palvelun saatavuuteen liittyvät tekijät. Redundanssin avulla, hajauttamalla palveluita eri palvelinkeskuksiin, palvelun toimintavarmuus kasvaa, koska yhden keskuksen vika, tai siihen kohdistuva palvelunestohyökkäys, ei vaikuta esimerkiksi verkkosivun toimintaan. Pilvipalveluissa tämä on helppoa, koska heillä on palvelinkeskuksia eri sijainneissa, ja saman palvelun tilaaminen useaan eri palvelinkeskukseen on tehty usein helpoksi. Pilvipalvelutkaan eivät ole täydellisiä, joten asiakkaan kannattaa perehtyä eri pilvipalveluntarjoajien antamaan informaatioon palvelun käyttökatkoista. (2, s. 27.)

Pilvipalveluntarjoajilla on palkkalistoillaan tietoturvaan keskittyviä työntekijöitä, joilla voidaan olettaa olevan parempi tietotaito ylläpitää turvallista palvelinympäristöä kuin henkilöllä, joka tekee päätyönään jotain muuta ja sivussa ylläpitää palvelinta. (2, s. 29.)

## 6.3 Pilvipalvelun heikkouksia

Pilvipalveluiden yhtenä suurena heikkoutena voidaan pitää sitä, että asiakkaan täytyy luottaa palveluntarjoajan antamiin suuriin lupauksiin tietoturvasta. Niiden todenmukaisuutta on lähes mahdoton selvittää, koska datakeskuksiin ei päästetä kaikkia asiakkaita tutustumaan. Asiakkaan kannattaakin keskittyä palveluntarjoajiin, jotka ovat suorittaneet jonkin standardin mukaisen turvallisuustodistuksen. Tosin myös niiden todellinen arvo asiakkaalle voi olla kyseenalainen, jos ei perehdy, mitä todistuksen saaminen vaatii, eivätkä nekään dokumentit yleensä ole kaikkien saatavilla. (2, s. 33–34.)

Varsinaiseen tietoturvaan liittyvä riski, jota paikallisella palvelimella ei ole, mutta pilvipalveluilla on, on resurssien jakaminen. Pilvipalvelussa useat asiakkaat käyttävät samaa fyysistä prosessoria ja muistia. Käyttäjien välinen raja lisää mahdollisia tapoja päästä käsiksi samaa resurssia käyttävien dataan. Resurssien jakaminen aiheuttaa myös sen, että palvelunestohyökkäys, kohdistuessaan yhteen asiakkaaseen, voi vaikuttaa saman resurssin palveluihin negatiivisesti. (2, s. 38–41.)

Muiden asiakkaiden lisäksi, myös pilvipalveluntarjoajan henkilökunta voi tarkoituksella tai vahingossa vaarantaa asiakkaidensa tietoturvan. Datan joutumista vääriin käsiin voi estää salauksella, mutta tahallisen palvelun toiminnan estämiseen on vain vähän keinoja. Tämän takia asiakkaan kannattaakin tutustua palveluntarjoajan mahdollisesti julkaisemiin henkilöstön tietoturvasääntöihin. (2, s. 46–47.)

Myös asiakkaan omat työntekijät ovat suuri riski. Pilvipalveluiden käyttö on yksinkertaista ja nopeaa verrattuna paikalliseen palvelimeen, jolloin uusia pilvipalveluita perustaessaan työntekijä voi vahingossa tai tahallaan avata uusia tapoja datan ulossaantiin tai tunkeutumiseen. (2, s. 51.)

Yksi mahdollinen uhkatekijä on pilvipalveluissa usein käytettävät käyttöjärjestelmien levykuvat. Periaatteessa on mahdollista jakaa levykuvia, jotka antavat pääsyn käyttäjän dataan (2, s. 47.) Suurin osa levykuvista on kuitenkin palveluntarjoajien itse jakamia, jolloin voinee olettaa tietoturvahaittojen määrän vähäiseksi. Niissäkin on mahdollisuus vahingossa jääneisiin reitteihin, mutta se ei eroa paikallisen palvelimen vaaroista.

Useissa valtioissa on lakeja, jotka mahdollistavat viranomaisten pääsyn dataan käsiksi. Tämän takia palvelinkeskusten sijainti voi vaikuttaa siihen, onko data oikeasti turvassa. Tämä ongelma koskee myös itse ylläpitämiä palvelimia, mutta pilvipalvelumalli, jossa data voi olla usean eri toimivallan alla, kärjistää ongelmaa. (2, s. 52.)

### 6.4 Tietomurron vaikutukset ja vaarat kalustonhallintajärjestelmässä

Jos järjestelmään pystytään vaikuttamaan internetin kautta, se voi aiheuttaa vaaroja tietomurron sattuessa. Parhaassa tapauksessa kalustoon ei pystytä vaikuttamaan, vaikka IoT-järjestelmään päästäisiinkin sisälle, mutta siinäkin tapauksessa dataa koneen toiminnasta ja mahdollisesti sijainnista joutuu vääriin käsiin. Mitään erityisen arkaluontoista dataa koneesta tuskin saadaan, tai tarvitsee näyttää nettisivulla. Projektissa IoT-järjestelmän vaatimuksiin kuuluivat datan esittämisen lisäksi parametrien asetus sekä tiedostojen lataus nettisivun kautta laitteelle. Nämä koneen toimintoihin vaikuttavat ominaisuudet voivat aiheuttaa vaaroja koneen operaattorille.

Parametrejä voivat olla esimerkiksi lämmittimien asetusarvot, tai puhaltimien oletusnopeus. Jos näitä sabotointimielessä muuttaa, ei koneen kuljettajalle aiheudu vaaraa, vaikka se tapahtuisikin koneen ollessa käytössä. Parametrien rajat täytyisi olla ohjainlaitteen puolella asetettuna niin, ettei kukaan voi vahingossakaan asettaa niitä vaarallisiin arvoihin, oli laite yhdistettynä verkkoon tai ei. Toinen menetelmä parametrien, kuten myös tiedostojen lähettämisen tekemiseksi turvallisemmaksi on se, että parametrit ja tiedostot otetaan käyttöön vasta, kun kone on pysähtyneenä, tai kun operaattori sen sallii. Muuten koneen toiminta voisi operaattorin kannalta olla ennalta-arvaamatonta, jos parametrejä muutetaan nettisivun kautta, mutta hänelle ei anneta ilmoitusta sen muutoksesta siinäkin tapauksessa, että se tehtäisiin sallitusti.

Tiedoston lähettämistä ohjainlaitteelle oli tarkoitus käyttää ohjainlaitteen päivittämiseen. Jos henkilö, joka haluaa aiheuttaa vahinkoa laitteelle, pääsee lähettämään sille omia ohjelmiaan ja tiedostoja, voi se muuttaa koneen toiminnan täysin. Tässä tapauksessa tunkeutujan täytyy tietää käytettävä ohjainlaite, kuinka siihen päivitetään ohjelmisto, millä se on ohjelmoitu sekä myös yrityskohtainen tapa, jolla päivitys otetaan käyttöön ohjainlaitteella. Todennäköisempi lopputulos väärällä tiedostolla päivittäessä olisi ohjainlaitteen toiminnan loppuminen. Turvallisuuden kannalta päivitys olisi paras aloittaa aina

operaattorin toimesta, laitevalmistajan ilmoituksen jälkeen. Tällöin ohjainlaite päivitettäisiin aina virallisella versiolla, mutta päivitystiedoston saisi kuitenkin lähetettyä laitteelle internetin välityksellä.

Jos konetta voisi ohjata tai käynnistää IoT-järjestelmän kautta, tapaturman riskit olisivat suuremmat kuin suunnitellussa järjestelmässä, jonka takia niiden tarpeellisuutta kannattaa harkita tarkasti. Sen takia näitä ominaisuuksia ei ollut järjestelmän vaatimuksissa.

# 7 Yhteenveto

Työn tarkoituksena oli selvittää tarvittavat sovellukset ja palvelut kalustonhallintajärjestelmän kehittämiseksi, ja tehdä prototyyppi siitä. Työssä valittiin tarvittavat palvelut pilvipalvelun puolelta ja verrattiin kolmen eri palveluntarjoajan hintoja. Työssä myös toteutettiin ja testattiin toteutus datan liikkeelle IoT-keskuksesta palvelimelle. Muita työn tavoitteita ei saavutettu ohjainlaitteen asennusongelmien ja tarvittavien taitojen puutteen takia. Suorittamatta jäivät datan siirto Codesys-suoritusympäristöstä Linux-järjestelmään ja sieltä pilvipalveluun. Datan siirto toisesta Linux-järjestelmästä, Raspberry Pi:stä onnistui kuitenkin. Myös datan visualisointi nettisivulla jäi tekemättä.

Työtä tehdessä selvisi, kuinka vaikeaa riisutun Linuxin kanssa työskentely on. Siihen ja visualisoinnin ja nettisivun toteuttamiseen paras ratkaisu voisi olla ulkopuolisten ohjelmoijien käyttö, jotta toteutuksesta tulisi vakaa ja järkevä. Järjestelmän kehittämiseen kuluva aika olisi todennäköisesti suuri, joten arvio sen kannattavuudesta täytyisi tehdä ennen kehittämisen jatkamista. Jos järjestelmän kehittämistä aiotaan jatkaa, seuraava askel voisi olla asiantuntijoiden löytäminen tehtäviin sekä arvioida yhdessä kehitykseen tarvittavaa työmäärää. Sen perusteella arvion tekeminen järjestelmän kannattavuudesta ja hinnasta asiakkaalle olisi helpompaa.

## Lähteet

1 Lea, Perry. IoT and Edge Computing for Architects - Second Edition. E-kirja. Packt Publishing.

2 Newcombe, Lee. 2012. Securing Cloud Services. United Kingdom: IT Governance Publishing.