UNIVERSITY OF OULU

TIETO- JA SÄHKÖTEKNIIKAN TIEDEKUNTA
TIETOTEKNIIKAN MAISTERIOHJELMA

Esineiden Internetin turvallisuusuhat ja protokollat

Tekijä	Jari Jussila
Valvoja	Kimmo Halunen
Toinen tarkastaja	Juha Röning

Marraskuu 2022

Jussila J. (2022) Esineiden Internetin turvallisuusuhat ja protokollat. Oulun Yliopisto, Tietotekniikan maisteriohjelma. Diplomityö, 59 s.

TIIVISTELMÄ

Esineiden Internet tarkoittaa elektronisia laitteita, jotka ovat yhdistetty Internetiin. Internetiin kytkettyjen laitteiden määrä on kasvanut nopeasti ja tämä on mahdollistanut uusia hyökkäysmenetelmiä hakkereiden käyttöön. Uudet hyökkäysmenetelmät aiheuttavat tietoturvauhkia, sekä uhkia käyttäjän yksityisyyteen. Jokainen IoT (Internet of Things)- laite tai IoT-sovellus, joka on yhteydessä verkkoon, käyttää tiedonsiirrossa jotain tiettyä protokollaa. Mainitussa IoT:ssa käytetään useita erilaisia protokollatyyppejä, joista tässä työssä laadittu taulukko. Taulukossa on vertailtu MQTT (Message Queue Telemetry Transport), DDS (Data Distribution Service), mDNS (Multicast Domain Name System) ja SSDP (Simple Service Discovery Protocol) protokollien keskeisiä ominaisuuksia. Tässä työssä on otettu MQTT tiedonsiirtoprotokolla tarkempaan tarkasteluun ja tutkittiin sen erilaisia ominaisuuksia. Ominaisuudet liittyvät toimintoon, jossa viestin lähettäminen ja vastaanottaminen tapahtuu välittäjän kautta. Yksi tietoturvallisuuteen liittyvä tarkastelun kohde oli viestien salaus, ja kuinka hyvin salaus toimii. Liikennöinnin ja suojauksen testaus voidaan suorittaa paikallisesti, eli tietokone voi kommunikoida sisäisesti eräänlaisen verkon kautta. MQTT:n ominaisuuksia testattiin tietokoneelle asennetulla Mosquitto välittäjäohjelmalla.

Ensimmäisessä testauksessa tutkittiin ja todennettiin viestin kulkemista välittäjän kautta vastaanottajalle, sekä tutkittiin kuittausviestien määrää eri palvelutasoilla. Teknisessä osuudessa testattiin, kuinka MQTT:ssä voidaan salata käyttäjän salasanan kryptaamista käyttämällä Mosquitto välittäjäohjelmaa. Samalla todennettiin myös pysyvän tallennuksen toimivuutta häiriötilanteen yhteydessä ja pohdittiin mahdollisia tallennettuun tietokantaan syntyviä tietoturvauhkia. Lisäksi testattiin viestiliikenteen salausta, käyttämällä porttia 8883, joka tukee TLS (Transport Layer Security) suojattua viestiliikennettä. Tuloksilla osoitettiin, että viestin rakenne muuttuu siten, että viestiä ei kyetä tulkitsemaan. Se koskee myös käyttäjätunnusta ja salasanaa. Testien tuloksilla haluttiin osoittaa, että MQTT:n ominaisuudet toimivat sille luvattujen standardien mukaisesti, jos Mosquitton konfigurointi on tehty oikein. Testituloksissa mm. QoS (Quality of Service) palvelutasoilla viestien datapakettien määrässä oli havaittavissa eroavaisuuksia. Testeissä havaittiin, että todennusyrityskertojen määrää ei ole rajoitettu MQTT:ssä. Lisäksi Wiresharkilla saaduissa testituloksissa ilmeni, että asiakkaan ja välittäjän välistä liikennettä on mahdollista lukea, mutta käytettäessä TLS suojausta sitä ei voida enää lukea. Testitulokset osoittivat todeksi sen, mitä teoriatekstissä oli kerrottu.

Avainsanat: Esineiden Internet, Mosquitto, MQTT, tietoturvallisuus, TLS

Jussila J. (2022) Internet of Things security threats and protocols. University of Oulu, Degree Programme in Computer Science and Engineering. Master's Thesis, 59 p.

ABSTRACT

The Internet of Things (IoT) refers to electronic devices that are connected to the Internet. The number of devices connected to the Internet has grown rapidly and this has enabled new attack methods for hackers to use. New attack methods cause information security threats, as well as threats to user privacy. Every Internet of Things device or application that is connected to the network uses a specific protocol for data transfer. In the mentioned IoT, several different protocol types are used, of which the table prepared in this work. The table compares the key features of the MQTT (Message Queue Telemetry Transport), DDS (Data Distribution Service), mDNS (Multicast Domain Name System) and SSDP (Simple Service Discovery Protocol) protocols. The features are related to the function where the message is sent and received through an intermediary. One subject of review related to information security was the encryption of messages, and how well the encryption works. Traffic and security testing can be performed locally, that is, the computer can communicate internally through a kind of network. MQTT features tested with the Mosquitto broker program installed on the computer.

In the first test, the passage of the message through the intermediary to the recipient was examined and verified, as well as the number of acknowledgment messages at different service levels was examined. The technical part tested how MQTT can be encrypted by encrypting the user's password using the Mosquitto broker program. At the same time, the functionality of the permanent storage was also verified in the event of a disturbance, and possible security threats to the stored database were considered. In addition, the encryption of message traffic was tested, using port 8883, which supports TLS (Transport Layer Security) protected message traffic. The results showed that the structure of the message changes in such a way that the message cannot be interpreted. It also applies to the username and password. The results of the tests were intended to show that MQTT's features work according to the promised standards if Mosquitto is configured correctly. In the test results, e.g. At QoS (Quality of Service) service levels, there were noticeable differences in the number of data packets of messages. The tests found that the number of authentication attempts is not limited in MQTT. In addition, the test results obtained with Wireshark showed that it is possible to read the traffic between the customer and the broker, but when TLS protection is used, it can no longer be read. The test results proved what was said in the theory text to be true.

Key words: Information security, Internet of Things, Mosquitto, MQTT, TLS

SISÄLLYSLUETTELO

1. JOHDANTO……8
2. ESINEIDEN INTERNET JA TIETOTURVAUHAT……10
    2.1. Esineiden Internet verkko ja IoT-verkkoliikenne……11
    2.2. Tietojärjestelmien ja IoT-ympäristön uhat……12
    2.3. IoT-verkoston puutteet ja laitteiden suojaaminen……14
    2.4. IoT-tietoturvavaatimukset……15
3. ESINEIDEN INTERNETIIN SISÄLTYVÄT PROTOKOLLAT……19
    3.1. SSL ja TLS-protokollien suojausmenetelmät……19
    3.2. DDS-protokolla ja havaitut haavoittuvuudet……21
    3.3. mDNS-palvelunetsintäprotokolla……22
    3.4. SSDP-verkkoprotokolla ja protokollan haavoittuvuudet……24
    3.5. MQTT-tiedonsiirtoprotokolla……25
        3.5.1. MQTT:n turvallisuusongelmat ja toimenpiteet……26
        3.5.2. Viestien lähetykset QoS-palvelutasoilla……27
    3.6. Eclipse Mosquitto-välittäjäohjelma……32
4. MQTT OMINAISUUKSIEN TODENTAMINEN……33
    4.1. Protokollatestauksissa käytetty laite ja järjestelmä……34
    4.2. Viestin lähetys ja viestiliikenteen mittaus……34
        4.2.1. Viestin lähetys QoS 0-palvelutasolta……34
        4.2.2. Viestiliikenteen mittaus palvelutaso QoS 0-käytöstä……36
        4.2.3. Viestin lähetys QoS 1 -palvelutasolta……37
        4.2.4. Viestiliikenteen mittaus palvelutasolla QoS 1……38
        4.2.5. Viestin lähetys QoS 2 -palvelutasolta……39
        4.2.6. Viestiliikenteen mittaus palvelutaso QoS 2 käytöstä……40
    4.3. Tietoturvan varmentaminen käyttäjätunnuksella ja salasanalla……42
    4.4. Haavoittuvuuksien todentaminen MQTT:ssä……43
    4.5. Viestiliikenteen suojaaminen TLS:n avulla……45
        4.5.1. TLS:n käyttämän liikenteen mittaaminen……47
        4.5.2. Viestien näkyvyys Wiresharkissa……49
    4.6. Viestien pysyvän tallennuksen merkitys……50
5. TESTIEN TULOKSET……52
    5.1. MQTT ja DDS-protokollien vertailu……53
6. POHDINTA……55
7. YHTEENVETO……56
8. LÄHTEET……57

ALKULAUSE

Arvoisa lukija, tässä diplomityössä on tarkoituksena tutkia esineiden Internet protokollien erilaisia ominaisuuksia, erityisen tarkastelun kohteena on MQTT tiedonsiirtoprotokolla. Se toimii IoT-laitteiden tiedonsiirtoprotokollana ja sitä voidaan testata tietokoneissa ja tietokoneeseen asennetuissa palvelimissa. Tässä työssä testataan ja analysoidaan viestin kulkua välittäjän ja kahden asiakkaan välillä. Testauksessa käytetään Mosquitto välittäjää käyttäen Windows komentokehotetta. Työssä perehdytään mm. MQTT protokollan viestiliikenteeseen.

Diplomityön pohjan rakentaminen alkoi syksyllä 2021, työn ensimmäisessä vaiheessa kerättiin teoriamateriaalia erilaisista verkkolähteistä ja tieteellisistä teksteistä, jotta teknisen osan laatiminen olisi mahdollista. Työn aiheeksi valikoitui tuli esineiden Internetin tietoturvallisuus, koska IoT-laitteiden käyttö on lisääntynyt Suomessa huomattavasti 2000-luvun aikana.

Liikenteen tarkkailua voidaan suorittaa omalla tietokoneella ja siihen asennetuilla tarvittavilla ohjelmilla. Testien suorittaminen ei vaadi ylimääräisiä rekisteröitymisiä erilaisiin palveluihin ja tarvittavat ohjelmat ovat ilmaisia. Testeissä kokeillaan käyttää porttia 8883, joka tukee TLS-standardien mukaisesti suojattua tietoliikennettä. Työ on vaatimustasoltaan sopivan haasteellinen. Kiitän Oulun Yliopistoa siitä, että diplomityö oli mahdollista tehdä.

Oulussa 21.11.2022

Jari Jussila

LYHENTEIDEN JA MERKKIEN SELITYKSET

AMQP	Advanced Message Queuing Protocol, avoin standardi sovelluskerroksen protokolla
API	Aplication Programming Interface, ohjelmointirajapinta
BGP	Border Gateway Protocol, reititysprotokolla
CSR	Certificate Signing Request, varmenteen allekirjoituspyyntö
DDS	Data Distribution Service, datakeskeinen standardiprotokolla
DdoS	Distributed Denial of Service, hajautettu palvelunestohyökkäys
DoS	Denial of Service, palveluestohyökkäys
DTLS	Datagram Transport Layer Security, tietoliikeprotokollaversio
EPL	Eclipse Public License, vapaiden ohjelmistojen julkaisemiseen tarkoitettu lisenssi
HTTPS	Hypertext Transfer Protocol Secure, hypertekstin siirtoprotokolla TLS/SSL suojauksella
IoT	Internet of Things, esineiden Internet verkko
IIoT	Industrial Internet of Things, verkkoon kytkettyjä teollisuuslaitteita
IP	Internet Protocol, verkkokerroksen protokolla
LAN	Local Area Network, lähiverkko
LoRaWAN	Low Range Wide Area Network, langaton verkkoteknologia
MQTT	Message Queue Telemetry Transport, viestintäprotokolla
M2M	Machine to Machine, laitteiden välinen suora tiedonsiirto
mDNS	Multicast Domain Name System, monilähetyksellinen palvelunetsintä protokolla
OMG	Object Management Group, tietokonealan standardien yhteenliittymä
QoS	Quality of Service, tietoliikenteen priorisointi ja luokittelu

RED	Radio Equipment Directive, radiolaitedirektiivi
SASL	Simple Authentication and Security Layer, Internet- protokollien todennus kehys
SFTP	SSH File Transfer Protocol, SSH-tiedostojen siirtoprotokolla
SSDP	Simple Service Discovery Protocol, pienissä verkoissa käytettävä verkkoprotokolla
SSL	Secure Sockets Layer, tietoverkkosalausprotokolla
TCP	Transmission Control Protocol, tietoliikenneprotokolla
TLS	Transport Layer Security, salausprotokolla
UDP	User Datagram Protocol, yhteydetön tietoliikenneprotokolla
UPnP	Universal Plug-and-Play, verkkopalveluja tukeva arkkitehtuuri
VPN	Virtual Private Network, käytetään verkkoliikenteen salaamiseen
WLAN	Wireless Local Area Network, langaton lähiverkko
WSN	Wireless Sensor Networks, langattomat anturiverkot

1. JOHDANTO

Tietoturvallisuus on aiheena mielenkiintoinen, koska se on tällä hetkellä erittäin ajankohtainen. Sitä todistavat viimeisen vuoden aikana lisääntyneet tietomurrot ja palveluestohyökkäykset. Tulevaisuuden tietoturvauhkista ei taida kenelläkään olla tarkkaa tietoa. Niinpä varautuminen uhkien torjumiseen kannattaa, sillä se voi tuoda yllättäviä haasteita. Tietoturvallisuus on aiheena hyvin laaja ja käsittää erilaisia aihealueita. Työtä oli syytä rajata käsittämään IoT-laitteiden tietoliikennettä ja siinä ilmeneviä tietoturvaan liittyviä uhkia. IoT-laitteet käyttävät useita erilaisia protokollia, joten aihetta täytyi rajoittaa lisää keskittymällä MQTT protokollaan.

Työn tavoitteena oli selvittää MQTT protokollan, lähinnä tietoturvaan liittyvien ominaisuuksien toimivuutta viestiliikenteessä. Lisäksi tutkittavana on tietoturvahaavoittuvuuksia, joita voi esiintyä protokollassa. Työssä käytettiin välittäjänä Mosquitto brokeria, joka käyttää viestiliikenteessä MQTT protokollaa ja tukee TLS:n käyttöä.

Työtä varten on laadittu tärkeitä tutkimuskysymyksiä, joilla halutaan selventää toteutetun työn etenemistä ja lopussa esitettyjen tuloksia ja niiden merkityksiä esimerkiksi tietoturvallisuuden osalta. Seuraavaksi esitellään tutkimuskysymyksiä ja niihin vastataan selkeästi.

- Miten palvelutaso näkyy viestiliikenteessä?
- Onko viestin pysyvä tallennus tietoturvauhka?
- Miten MQTT:ssä asiakkaan salasana ja viesti salataan?
- Miten viestin kohdentaminen halutulle asiakkaalle parantaa tietoturvaa?

Diplomityön toisessa kappaleessa käsitellään aluksi tiivistetysti esineiden Internet verkon kehittymistä ja elektroniikkalaitteita, jotka luokitellaan IoT-laitteiksi. Kappaleessa mainitaan myös sensoreiden liittämisen helpottumisesta Internetin verkkoprotokollien suhteen. Samalla kerrotaan IoT-järjestelmien muodostumisesta langattomista ja kiinteistä verkoista. Työssä käsitellään lyhyesti myös esineiden Internetin ja teollisuuden esineiden Internetin eroavaisuuksia. Kappaleessa 2 käsitellään myös IoT-automaatiojärjestelmien haavoittuvuuksia, uhkia ja riskejä, jotka ovat kyberturvallisuuden kannalta kolme merkittävää käsitettä. Samassa kappaleessa käsitellään erilaisia tietoturvavaatimuksia ja IoT- arkkitehtuuria. Samoin käsitellään myös pilvipalveluiden suojausmenetelmiä ja miksi kyseiset suojausmenetelmät ovat tärkeitä, sekä mahdollisia ongelmia, joita suojausmenetelmät voivat aiheuttaa.

Kolmannen kappaleen aiheet ovat IoT-protokollat, sekä eri protokollien merkitykset tietoturvahaavoittuvuuksissa, joita voi protokollissa esiintyä. Työn laajuuden rajoittamista varten, työssä keskitytään tarkastelemaan tarkemmin MQTT-protokollan erilaisia ominaisuuksia. MQTT-protokolla valittiin testikohteeksi, koska se on monipuolinen ja soveltuu erilaisiin IoT-systeemeihin. Lisäksi kappaleen kolmen aiheeseen sisältyy myös mm. MQTT-protokollan palvelutasot, sekä MQTT:ssä ilmeneviä haavoittuvuuksia.

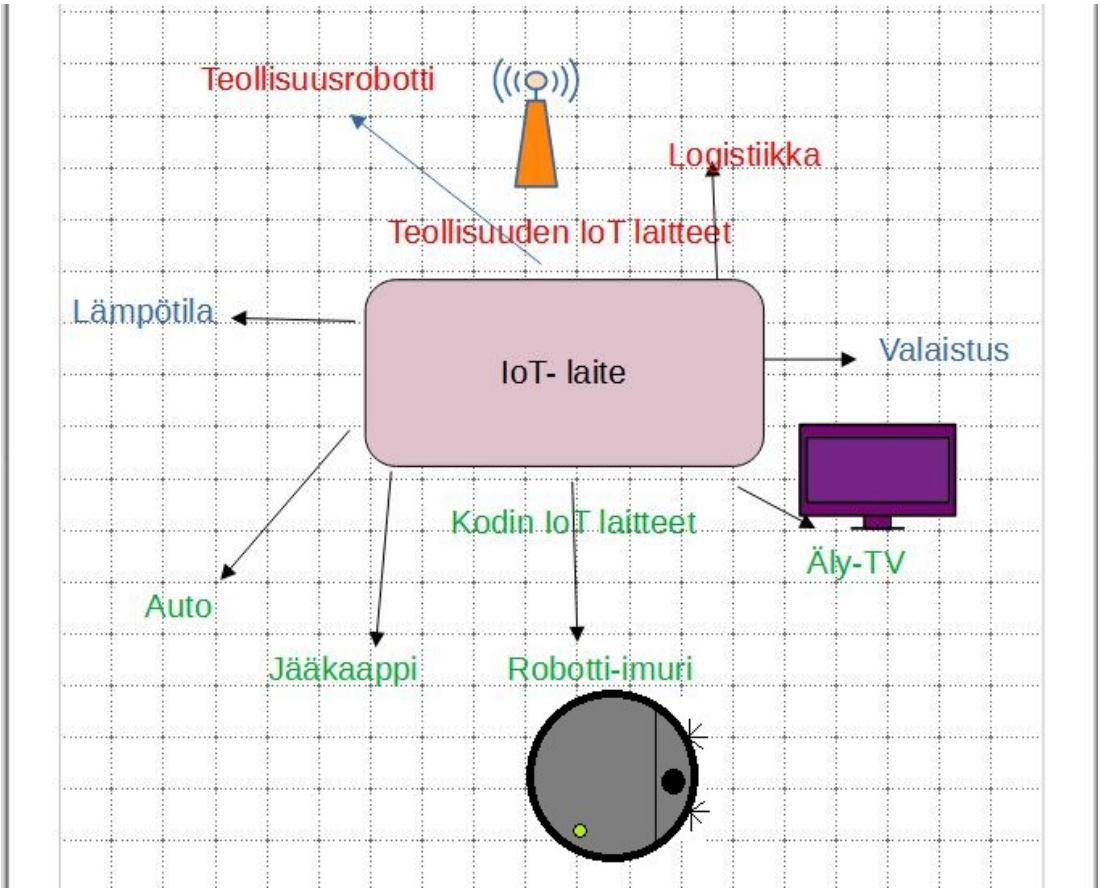
Neljännessä kappaleessa käsitellään protokollien ominaisuuksien testaamista, sekä samalla vastataan tutkimuskysymyksiin tulosten perusteella. Teknisessä osassa tutkitaan myös verkkoliikennettä ja siihen vaikuttavia tekijöitä, kuten palvelutasoja. Lisäksi tarkastellaan käyttäjätunnusten todentamista ja siihen liittyviä haavoittuvuuksia. Testissä testataan myös TLS-salausta, joka käyttää siihen varattua porttia 8883. Edellä mainittuja ominaisuuksia tutkittiin Wireshark-verkko analysaattoriohjelmalla. Tutkimuskysymyksissä on maininta myös viestien pysyvästä tallennuksesta välittäjään. Pysyvän tallennuksen lisäksi tutkitaan tallennuksen aiheuttamia mahdollisia haavoittuvuuksia. Viidennessä luvussa käydään läpi tuloksia, joita on saatu testien aikana. Tuloksissa tarkastellaan tietoturvaan liittyviä kysymyksiä viestiliikenteen mittauksissa, sekä TLS:n käyttöä viestiliikenteen salauksen turvaamiseksi. Samassa kappaleessa on laadittu vertailutaulukko, jossa on vertailussa MQTT- ja DDS-tiedonsiirtoprotokollat. Vertailutaulukon tarkoitus on osoittaa kahden tiedonsiirtoprotokollan eroavaisuutta.

2. ESINEIDEN INTERNET JA TIETOTURVAUHAT

Tietoturvassa on yleisesti ottaen kolme tärkeää käsitettä, luottamuksellisuus, eheys ja saatavuus. Luottamuksellisuus tarkoittaa, että ulkopuolinen ei pääse käsittelemään arkaluontoisia tietoja. Arkaluontoisia tietoja tulee päästä käsittelemään vain ne henkilöt, joille on myönnetty käyttöoikeudet kyseisiin tietoihin. Mikäli tietoja päätyy vääriin käsiin, seuraamukset voivat olla erittäin ikäviä [1 s. 29]. Tietoturvallisuudesta huolehtimiseen kuuluu myös muita toimenpiteitä, kuten tietojen pysyminen ehyinä. Tietojen eheydessä on tärkeää, että tiedot säilyvät muuttumattomina. Tietoja saa muuttaa vain tietyt käyttäjät, joilla on käyttöoikeudet muokata tiedon sisältöä tietyillä ehdoilla [1 s. 29-30]. Tiedon saatavuudella tarkoitetaan, että tieto on hyödynnettävissä haluttuna aikana, kuten pankkijärjestelmän toiminnoissa [1 s. 30]. Turvatoimenpiteisiin kuuluu myös anonymiteetin säilyttäminen ja pääsynhallintaluettelon ylläpitäminen. Lisäksi yksityisyyden merkitys IoT:n elinkaarenaikana täytyy olla asianmukainen [2 s. 4-5].

IoT teknologia nousi varsin tärkeään asemaan 2000-luvulla, koska jokapäiväiset laitteet mahdollistivat kytkeytymisen Internetiin sulautettujen laitteiden kautta. Sulautetut laitteet antavat mahdollisuuden saumattomaan viestintään ihmisten, prosessien ja asioiden välillä [3]. Arkielämä muuttuu nopeasti automaattisemmaksi, koska älylaitteiden valikoima monipuolistuu ja IoT-laitteita on nähtävillä joka paikassa. Ainutlaatuisten pilvipalveluteknologioiden määrä kasvaa kiihtyvällä tahdilla, josta yhtenä esimerkkinä on digitaalisen elintarvikelistan saanti älyjääkaapeista ja Internetin avulla pilvipalvelimiin kytketyt 3D-tulostimet, joiden avulla voidaan tulostaa lähes mitä tahansa materiaalia [4 s. 18].

Teollisuudelle on kehitetty erikseen Industrial IoT (IIoT), eli teollinen esineiden Internet. Teollisuudelle kehitetyt IIoT-laitteistot sisältävät antureita, joita tarvitaan kokoonpanolinjalla tai muissa valmistusprosessien valvonnassa. Nuo anturit keräävät erilaisia tietoja, jotka välitetään niille tarkoitetuille valvontasovelluksille ja ne varmistavat, että keskeiset prosessit toimivat optimaalisesti [5]. Lisäksi logistiikan alalla käytetään IoT:tä ja niiden käyttäminen tulee kasvamaan tulevaisuudessa. Jotta logistiikka toimii, on IoT:tä käytettävä myös liikenneinfrastruktuurin kehittämiseen ja parannettava valvonta- ja seurantajärjestelmää liikenteessä [6].



Kuva 1. Esimerkkejä kuluttajien ja yritysten IoT-laitteista ja teollisuuden IIoT-laitteista.[1]

2.1. Esineiden Internet verkko ja IoT-verkkoliikenne

IoT-verkkojen tavoitteena on yhdistää ja vaihtaa tietoja eri laitteiden ja järjestelmien välillä Internetin välityksellä. IoT-laitteiksi luokitellaan elektroniset laitteet, jotka ovat yhteydessä tietoverkkoihin [7]. J. Cynthia et al. mukaan IoT-laite sisältää myös sulautetun ohjelmiston [2 s. 1]. IoT-laitteiden ominaisuutena on tietojen jakaminen ja tietojen vastaanottaminen. Edellä mainitusta johtuen IoT-laitteilla on mahdollisuus toimia saamiensa tietojen perusteella itsenäisesti, tai voivat toimia osana suurempaa järjestelmäverkostoa [7]. IoT-järjestelmä koostuu teknillisesti langattomista ja kiinteistä verkoista, joihin laitteet ovat yhteydessä langattomalla tai kiinteällä yhteydellä. Laitteiden määrän lisääntyessä on ollut tärkeää kehittää myös langattomia verkkoteknologioita. Lisäksi Empirican artikkelissa mainitaan [7], että tuoreimpana langattomana verkkotekniikkana on 5G-verkko, joka on nimenomaan suunniteltu IoT-laitteiden tarpeita varten. 5G-verkot mahdollistavat korkeammat kapasiteetit lyhyillä viiveillä, mutta IoT-laitteet toimivat myös 3G- ja 4G-verkoissa. IoT-laitteet on mahdollista kytkeä langattomaan LoRaWAN-verkkoon (Low Range Wide Area

1 Alkuperäinen kuva osoitteessa https://internetofthingsagenda.techtarget.com/definition/IoT-device.

Network), jonka teknologia pohjautuu LoRa-teknologiaan. LoRaWAN verkko on käytännöllinen, koska sitä voidaan käyttää liikkuvissa kohteissa. Suomesta löytyy LoRaWAN-verkko, jonka ylläpitäjänä toimii Digita [7].

Tavallisen kuluttajan kohdalla voidaan pitää IoT:tä kuitenkin melko uutena asiana, vaikka lähes samanlaista automaatiotekniikkaa on käytetty eri aloilla useamman vuosikymmenen ajan. Esimerkkinä voidaan mainita Reijo Ratauoman säätiön julkaisussa M2M (Machine to Machine) tekniikka. Kyseistä tekniikkaa käytetään kiinteistöissä sähkömittareiden etälukemista varten [6].

Vaikka IoT-tekniikka on ollut olemassa jo kauan, teknologian osalta käytännöllisyys on kehittynyt eteenpäin varsin nopeasti viime aikoina. Käytännöllisyys on johtanut anturitekniikan valmistuskustannusten halpenemiseen ja se puolestaan on edesauttanut IoT-tekniikan päätymistä useimmille laitevalmistajille. Antureiden yhdistämistä Internetin verkkoprotokolliin on helpotettu, jotta tiedonsiirto olisi huomattavasti tehokkaampaa [3]. IoT-elektroniikkalaitteissa esiintyy paljon erilaisia antureita, joiden avulla saadaan tuotettua erilaista dataa, joten muutamissa sekunneissa dataa saattaa kertyä todella paljon. IoT-sovellusten kehittäminen helpottuu ja nopeutuu sitä mukaan, kun IoT-palvelimet, anturilaitteet ja muut IoT-alustat kehittyvät tekniikan osalta [6].

IoT-laitteet on suunniteltu keräämään, analysoimaan ja tallentamaan dataa ilman käyttäjän antamia suoria syötteitä. IoT-laitteella ei ole useinkaan tietokoneen tai älypuhelimen sisäänrakennettua tietoturvaa. IoT-laitteella voi kuitenkin olla yhteys yrityksen kriittisiin järjestelmiin. IoT-laitteet voivat myös aiheuttaa riskin pilvipalveluiden tasolla. On arvioitu, että 90 prosenttia IoT-laitteista tulevasta datasta virtaa pilvipalvelujen kautta, joista valtaosa jakaa palvelin- ja tallennuslaitteistoja pilvipalvelun asiakkaiden kesken. Pilvipalvelijoille tärkeintä on luottamuksellisuus, sekä vahva ja jatkuva tiedonsalaus, jotka tekevät pilvipalveluista turvallisempia [8 s. 5].

2.2. Tietojärjestelmien ja IoT-ympäristön uhat

J. Cynthia et al. toteavat julkaisussaan, että tietojen ollessa saatavilla kaikkialta, se tekee tiedoista haavoittuvia tietoturvauhkille ja -hyökkäyksille. Lisäksi J. Cynthia et al. toteavat, että IoT-verkon vaarantunut osio tekee IoT-verkon muista kokonaisuuksista haavoittuvia [2 s. 2]. Järjestelmissä esiintyvät haavoittuvuudet antavat hyökkääjälle mahdollisuuden käyttää järjestelmää tavalla, jollaiseen sitä ei ole tarkoitettu [1 s. 32-33]. Suomalainen tietoturvayhtiö F-Secure on tehnyt raportin fyysisten laitteiden haavoittuvuuksista [9]. Yhtiön raportissa kerrotaan kuinka uhkan voi toteuttaa käyttämällä kahta väärennettyä verkkokytkintä. Niiden tavoitteena on ohittaa järjestelmäkomponenttien todentajaprosesseja [9]. Testitulokset osoittivat, että laiteväärennöksiä löytyi sitä mukaa, kun väärennetyt laitteet lakkasivat toimimasta uusien ohjelmistopäivitysten myötä. F-Secure oli myös havainnut, että

verkkokytkinlaiteväärennökset olivat nimenomaan kehitetty ohittamaan todentajien toimenpiteitä [9].

Tietoturvahaavoittuvuuden vakavin muoto on nollapäivähaavoittuvuus joka on tavallinen haavoittuvuus, mutta sitä varten ei ole vielä julkaistu suojauspäivitystä. Tämä tarkoittaa sitä, että järjestelmät ja ohjelmistot ovat edelleen haavoittuvaisia, vaikka päivitykset olisi asennettu. Toinen uhkamuoto on julkisuudelle täysin tuntemattomat haavoittuvuudet [1 s. 32-33].

Tietoturvallisuuden käsite uhka kohdistuu joko organisaation sisäpuolelle tai ulkopuolelle. Sisäpuolelta tulevilla uhkilla on suuria vaikutuksia talouteen, maineeseen tai tietopääomaan. Sisäisten uhkien lähteinä ovat usein tyytymättömät työntekijät, alihankkijat tai yhtiökumppanit [4 s. 87-88]. Ulkoisia uhkia ovat organisaation ulkopuolelta tulevat uhkatekijät, kuten mm. erilaiset haittaohjelmat, madot, virukset jne. Myös kybersota, kyberterrorismi ja haktivismi ovat ulkoisia uhkia [4 s. 91-92].

J. Limnéll et al. toteavat, että käsitettä uhka voidaan vielä torjua, mutta toista käsitettä eli riskejä ei voida, koska riskit ovat kytköksissä kaikkiin toimintoihin, erityisesti tietoturvamaailmassa. Riskillä tarkoitetaan jotain negatiivista tapahtumaa, joka voi tapahtua tulevaisuudessa [4 s. 88]. Riskien kartoituksessa arvioidaan mahdollisten negatiivisten tapahtumien todennäköisyyttä ja mitkä ovat tapahtumien vaikutukset. Jos edellä mainittuja uhkatapahtumia ei voida arvioida, joudutaan toimimaan epävarmuudessa. Epävarmuus voi johtaa yleensä suojaustoimenpiteiden yli- tai alilyönteihin, sekä resurssien hukkaamiseen [4 s. 88-89].

J. Cynthia et al. esittävät julkaisussaan ohjeita IoT:n suojausvaatimusten toteuttamiseksi. IoT-laitteiden suojausvaatimuksissa otetaan huomioon järjestelmissä sisältävien porttien pääsynhallinta. Lisäksi IoT-laitteiden käyttöjärjestelmät suojataan rajoitetuilla käyttöoikeuksilla. Suojauksen varmistamiseksi käytetään vaiheittaista käynnistysjärjestystä ja jokaisessa käynnistysvaiheessa tarkistetaan käynnistyskoodin kelvollisuus [2 s. 3]. Jos IoT-laitteet on kytketty reitittimien kautta yleiseen dataverkkoon, on syytä tarkistaa laitteen salasanat ja muut suojaustoimet. Tämän avulla parannetaan tietoturvaa. Tietoturvan kannalta tietoliikenneverkkolaitteet keräävät lokitietoja, joiden avulla voidaan selvittää erilaisia ongelmatilanteita. Lisäksi laitteet tallentavat lokitietoja, mutta tietojen tallentamisessa ongelmana on, että lokitiedot sisältävät yksityisyyden suojaan liittyvää dataa [1 s. 159]. Nykyään voidaan kuvata ihmisen ja IoT-laitteiden välistä suhdetta älykkääksi ja vuorovaikutteiseksi, joissa määrittyy ihmisten ajankäyttö, toiminnat ja ajattelutavat. Tästä nähdään teknologian nopea kehittyminen, joka on tapahtunut yhden sukupolven aikana [4 s. 23].

2.3. IoT-verkoston puutteet ja laitteiden suojaaminen

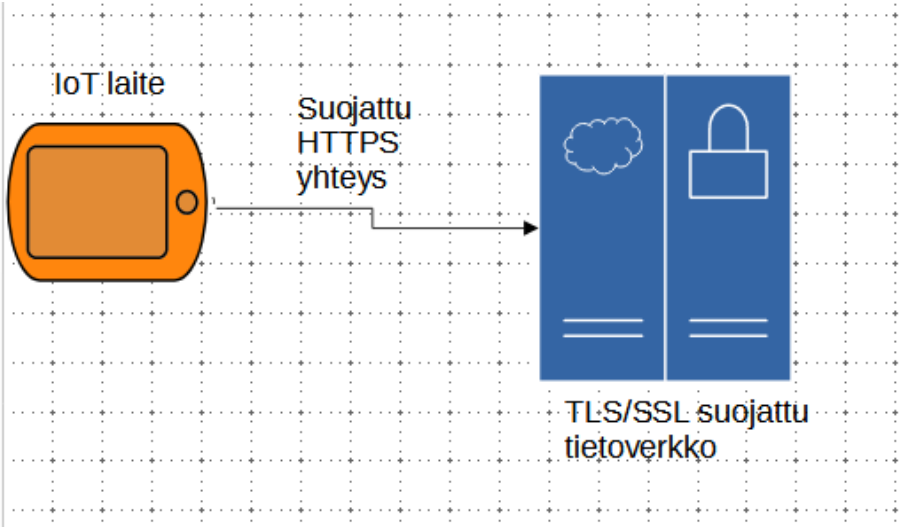
Kyberturvallisuuskeskuksen teettämässä vuosikatsauksessa kävi ilmi, että älylaitteet olivat tietoturvan suhteen vielä puutteellisia ja varsinkin silloin, kun laitteet olivat kytkettyinä tietoverkkoihin. Vuosikatsauksesta kävi ilmi, että jopa puoli miljoonaa tulostinta olivat suojaamattomia ja puolet niistä oli saatu testeissä kaapattua, jolloin niillä oli mahdollista tulostaa [10 s. 15].

IoT-verkko voi haavoittua kokonaisuudessaan, jos yksikin verkon entiteeteistä vaarantuu. Tietopalvelut voivat joutua tietoturvauhkien kohteiksi. Sama uhka koskee myös WSN (Wireless Sensor Networks) anturiverkkoja, Internetiä ja IoT-verkkoja. IoT-verkkojen puutteet altistavat ulkoisille hyökkäyksille ja muille uhkille, jos laitteilla ei ole vahvaa autentikointia, IoT-tietojen salausta ja salausavainten hallintaa [2 s. 1-2]. IoT-laitteet on mukautettava toimimaan Internetissä. Niissä on usein myös rajoitettu laskentateho ja muistikapasiteetti, minkä vuoksi salausalgoritmien käyttö on haasteellista. Salausalgoritmit vaativat yleensä myös runsaasti resursseja [2 s. 18].

IoT-maailmassa ollaan kuitenkin tekemässä paljon muutoksia. Kyberturvallisuuskeskuksen julkaisemassa vuosikatsauksessa 13/2021 on mainittu, että EU on säätänyt joitakin vuosia sitten kyberturvallisuusasetuksessa uuden eurooppalaisen tietoturvallisuus sertifiointijärjestelmän, jonka johdosta lähivuosina laaditaan IoT- laitteille omat sertifiointijärjestelmät [10 s. 15].

Lainsäädännön avulla voidaan parantaa verkkoon kytkettyjen laitteiden tietoturvallisuuteen liittyvää asemaa. Euroopan Unionin komissio on jo huomioinut tämän. Traficomin vuosikatsauksessa on mainittu, että turvallisuuden parantamiseksi on ehdotettu EU:ta varten omaa radiolaitedirektiivin, Radio Equipment Directive (RED) säätämistä. Säädöksellä on tarkoitus mullistaa IoT-maailmaa merkittävästi [10 s. 15].

Vahvan salasanan käyttöä suositellaan erilaisissa verkkopalveluissa ja samaa voidaan suositella myös IoT-laitteille. Kyberturvallisuuskeskuksen mukaan vahvan salasanan perustana on isojen ja pienten kirjainten käyttö, sekä numeroiden ja erikoismerkkien käyttöä. Tällä tavoin salasanasta tulee vahvempi ja sen murtaminen on vaikeampaa. Lisäturvaa tuo se, että jokaisessa palvelussa ja laitteissa on eri salasana, jolloin tietomurtojen riski pienenee huomattavasti. Kyberturvallisuuskeskus suosittelee lisäksi salasanojen hallintaohjelman käyttöä [11]. Kuvassa 2 esiintyy IoT, joka ottaa suojatulla yhteydellä yhteyden suojattuun tietoverkkoon/tietokantaan, jossa käytetään TLS/SSL suojausta.



Kuva 2. Yksittäinen laite on linkitetty tietoverkkoon, jossa on käytössä TLS/SSL suojaus. [2]

2.4. IoT-tietoturvavaatimukset

Esineiden Internetin tietoturvallisuudessa on tärkeää huomioida monia tietoturvavaatimuksia ja ne on syytä aina tarkistaa IoT-laitteiden kohdalla koko niiden elinkaarien kaikissa vaiheissa. Tällä lievennetään IoT-järjestelmiin kohdistuvia hyökkäyksiä. Merkittäviä tietoturvavaatimuksia ovat ainakin salausalgoritmit, joissa symmetriset algoritmit ovat kevyitä, verrattuna epäsymmetrisiin algoritmeihin. Sen vuoksi symmetrisiä algoritmeja suositellaan tiedonsiirron turvaamiseen [2 s. 2].

J. Cynthian et al. mainitsevat, että tietojen luokittelua pidetään tärkeänä tietoturvavaatimuksena. Lisäksi parannettaessa tietoturvallisuutta IoT-laitteiden käyttöjärjestelmät tulee olla suojattuja, samoin tulee rajoittaa käyttöoikeuksia ja hälventää laitteiden näkyvyyttä. Suunniteltaessa IoT-käyttöjärjestelmää, on hyvä käyttää järjestelmän luomiseen vain tarvittavia komponentteja, paketteja ja kirjastoja [2 s. 2-3].

Tehostettaessa tietoturvallisuutta, tulee huomioida luottamuksen hallinta. Luottamuksen hallinnalla tarkoitetaan viestinnän kokonaisuutta käyttäjien välillä. Luottamuksen hallintaa tarvitaan tietojen todentamisen, keräämisen ja hävittämisen vaiheissa [2 s. 5]. Tietoturvavaatimuksena on suositeltavaa käyttää vahvoja salaustekniikoita tai digitaalisia allekirjoituksia. J. Cynthian et al. esittävät julkaisussaan, että esineiden Internetin erilaisten verkkojen yhteenliittyminen on usein käytössä oleva rakenne, vaikka sillä on omat yhteensopivuus- ja tietoturvaongelmat. Erilaisten verkkojen ympäristössä tulee vaikeuksia tunnistaa luotettuja solmukohtia [2 s. 4-5].

2 Alkuperäinen kuva osoitteessa https://wiki.metropolia.fi/display/alykas/Tietoturvan+merkitys+ja+periaatteet+IoT%3Assa.

IoT:n tietoturvavaatimuksiin kuuluu myös IoT-verkkojen suojauksien vahvistuttaminen. Suojauksessa on tärkeää käyttää vahvimpia ja uusimpia salausmenetelmiä [2 s. 4]. Lisäksi J. Cynthia et al. toteavat, että IoT-laitteiden liitäntöjen määrä tulee olla minimitasolla, koska liitettyjen laitteiden kautta saadaan yhteys ulkoiseen verkkoon. Pääsy laitteen käyttämiseen tulee tapahtua vain rajoitetuista porteista, rajapinnoista ja tiettyjen palveluiden kautta. Yhteydet tulee suojata suojausprotokollilla, kuten HTTPS (Hypertext Transfer Protocol Secure) ja SFTP (SSH File Transfer Protocol) [2 s. 4].

Esineiden Internetin arkkitehtuurissa on käytössä erilaisia IoT-protokollapinoja, joista useimmat koostuvat IoT-suojausprotokollista. IoT-protokollapinot on suunniteltu toimimaan useissa eri kerroksissa antaen tietoturvallisuustakuun. Fyysisessä protokollakerroksessa on otettava huomioon teho, kaistanleveys ja energiankulutus. Tähän kerrokseen kiinnitetyt laitteet altistuvat erilaisille tietoturvahaasteille, kuten laitteiden fyysiseen peukalointiin, salakuunteluun ja tietojen muuttamiseen. Fyysisessä protokollakerroksessa salausalgoritmeilla on tärkeä rooli tietoturvallisuuden kohdalla [2 s. 8-9].

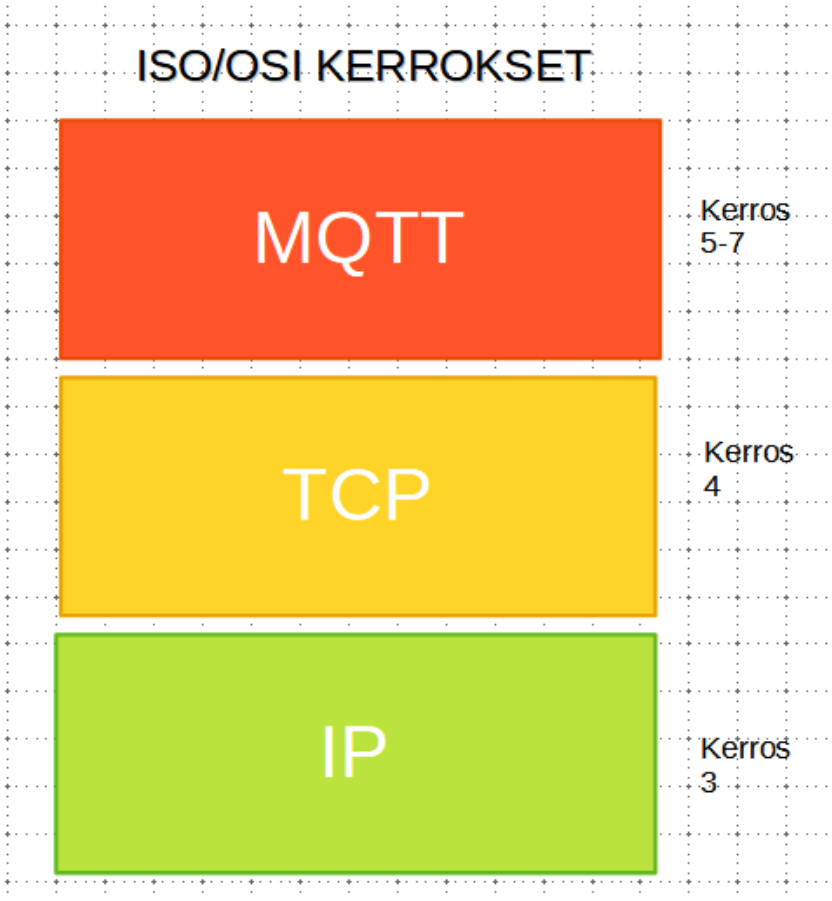
J. Cynthia et al. mukaan tiedon yhdistämiskerroksessa teholisäyksen avulla saadaan nostettua tiedonsiirtonopeutta langattomassa viestinnässä. Verkkokerroksessa on käytössä yleensä 6LoWPAN- ja IPSEC-protokollat, jotka huolehtivat kerroksen suojaamisesta. 6LoWPAN käyttää rajoitettuja solmuja ja rajoittamattomien laitteiden IoT-osoitteissa on käytössä IPv6 [2 s. 10]. Sovelluskerroksen ratkaisuna on käytetty jo varsin pitkään HTTP-protokollaa, jota käytetään Internetissä. HTTP-protokolla on kätevä tietojen hankkimisessa pyyntö- ja vastausmalleissa. Lisäksi IoT-laitteet siirtävät tietoja pilvipalvelimelle tai tavallisille palvelimille [2 s. 22].

TCP (Transmission Control Protocol) on laajassa käytössä oleva kuljetusprotokolla, vaikka sen luotettavuus ei ole kaikkein varminta. Toimintaperiaatteeltaan TCP toimii kolmisuuntaisella kättelyprosessiperiaatteella. TCP toimintaprosessin vaiheisiin kuuluvat yhteydenmuodostusvaihe, tiedonsiirtovaihe ja yhteyden katkaisuvaihe [2 s. 19-20]. TCP:tä pidetään pohjimmiltaan huomattavasti monimutkaisempana verrattuna UDP:hen (User Datagram Protocol), mutta taas UDP:n ongelma on sen epävakaus [2 s. 22]. UDP:llä voidaan lähettää datapaketteja aivan kuin TCP:llä, mutta UDP ei tarkista paketteja virheiden varalta. UDP:llä lähetetään paketteja vastaanottajalle, eikä lähettäjä edes välitä, että saako vastaanottaja paketit perille vai ei. Siitä huolimatta UDP jatkaa datapakettien lähettämistä. Mikäli vastaanottaja on unohtanut muutamia UDP-paketteja välillä, niitä ei voida pyytää uudelleen. Lisäksi kadonneita paketteja ei voi saada takaisin. UDP on siten kevyempi ja helppokäyttöisempi, kuin TCP, koska UDP:ssä on luovuttu pakettien perille menon varmistuksesta [12].

TCP toimii selaimelta lähetettävien pakettien numeroijana, jolloin vastaanottaja saa paketit oikeassa järjestyksessä. Vastaanottaessaan paketin, vastaanottaja lähettää kuittausviestin lähettäjälle. Mikäli lähettäjä ei saa kuittausviestiä, lähettäjä olettaa, että pakettia ei ole vastaanotettu ja lähettää paketin

uudestaan vastaanottajalle. TCP on tarkka luotettavuuden osalta, jolloin paketit tarkistetaan mahdollisten virheiden vuoksi ja samalla varmistetaan pyyntöjen täyttymisen oikeellisuus. TCP-pakettien perille menoa valvotaan tarkasti, jottei tietoja katoa matkan aikana [12] .

Lisäksi paketit tarkistetaan, että niihin ei sisälly korruptoituneita tietoja. Ladattaessa tiedostoja TCP:n avulla, verkossa voi esiintyä ongelmia siirron aikana, tästä huolimatta TCP-paketit pyritään vastaanottamaan täydellisinä. Järjestelmä voi kuitenkin lopettaa toimintansa, jos toinen osapuoli on täysin offline-tilassa. Järjestelmä antaa virheilmoituksen, jossa se ilmoittaa, että ei voinut määritellä viestintäkanavaa etäisännän kanssa [12].



Kuva 3. MQTT (Message Queue Telemetry Transport) protokollaa käytetään OSI-mallissa kerroksissa 5, 6 ja 7.[3]

3 Alkuperäinen kuva osoitteessa https://www.hivemq.com/blog/mqtt-essentials-part-3-client-broker-connection-establishment/.

Kuljetusprotokolla	UDP	TCP
Yhteys	Yhteydetön	Yhteyssuuntautunut
Jaksotus	UDP lähettää paketit ilman numerointia	TCP numeroi paketit, jotta vastaanottaja voi järjestää ne oikeaan järjestykseen
Nopeus	Nopea	Hidas
Luotettavuus	Matala	Korkea
Otsikon koko	Paketit ovat kevyitä, samalla otsikot ovat mahdollisimman minimaalisia	Paketit ovat kooltaan suuria, koska ne sisältävät useita kenttiä
Virheiden havainnointi/korjaus	Paketit tarkistetaan virheiden varalta, mutta ei korjaa virheitä. Vialliset paketit hylätään suoraan, eikä niitä enää pyydetä uudestaan	Paketit tarkistetaan virheiden varalta. Virheelliset paketit korjataan
Kuittaus	Ei kuittausta paketin lähetyksestä	Vastaanottaja kuittaa lähetetyn paketin
Siirtomenetelmä	Yksittäiset paketit	Kaikki paketit
Ruuhkan hallinta	Ei	Kyllä
Ohjelmistot	Videoneuvottelut, pelit, lähetykset	Tiedon siirto, sähköposti, verkkoselain

Taulukko 1. Vertailussa TCP- ja UDP-kuljetusprotokollat.[4]

4 Alkuperäinen taulukko osoitteessa https://www.vpnmentor.com/blog/tcp-vs-udp/.

3. ESINEIDEN INTERNETIIN SISÄLTYVÄT PROTOKOLLAT

IoT-laitteiden protokollissa on paljon eroavaisuuksia monestakin eri näkökulmista, muun muassa arkkitehtuuri- ja vuorovaikutusmallit, sekä kuljetusprotokollat. Tietyt protokollat on kehitetty pääasiassa asiakas-/palvelin -arkkitehtuurissa, ja jotkut taas kehitetty hajautettuun arkkitehtuuriin [13 s. 3]. Asiakas-/palvelinarkkitehtuuria käytetään protokollina mm. MQTT (Message Queue Telemetry Transport) [14] ja AMQP (Advanced Message Queuing Protocol) [15], jotka ovat yhteydessä asiakkaisiin vastaanottamalla ja välittämällä viestejä [13 s. 3].

3.1. SSL ja TLS-protokollien suojausmenetelmät

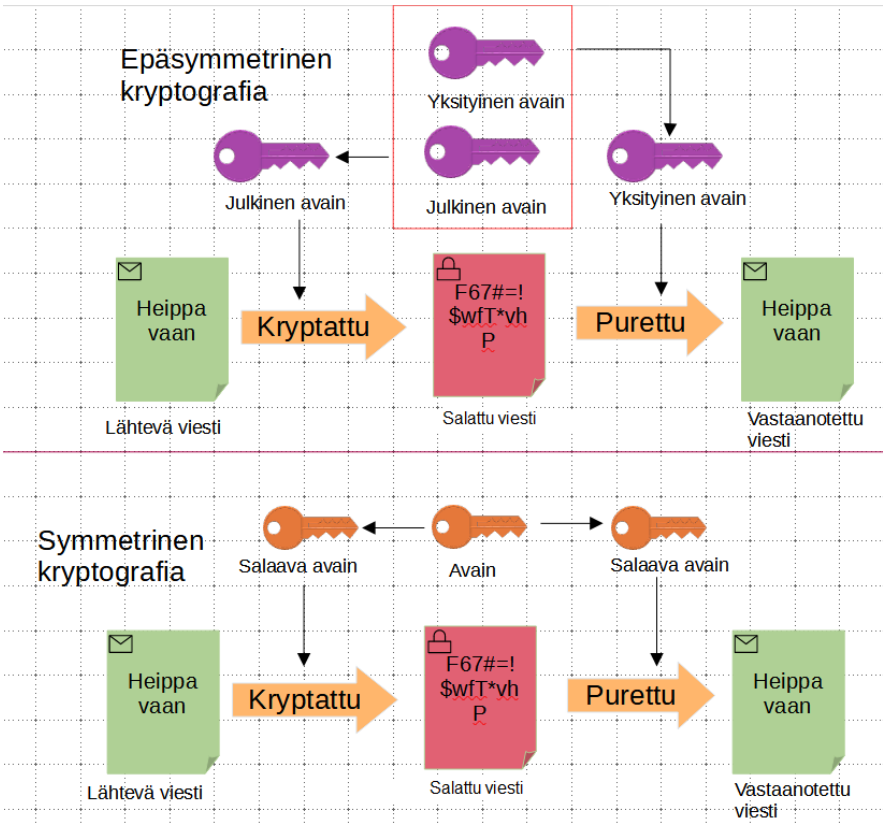
J. Cynthia et al. mukaan TCP protokollan suojausta voidaan lisätä käyttämällä SSL:ää (Secure Sockets Layer) tai uudempaa TLS (Transport Layer Security) suojausta. MQTT käyttää TCP:tä viestinnässä yhteyksien muodostamiseen, tiedonsiirtoon ja yhteyden katkaisuun [2 s. 19-20]. A. Viswanathan toteaa julkaisussaan, että MQTT ei käytä salattua viestintää. Salaamattoman viestinnän suhteen MQTT luottaa TCP-arkkitehtuuriin. MQTT kuitenkin tarjoaa mahdollisuuden lisäturvan tuomiseen TLS:n avulla. A. Viswanathan toteaa, että TLS:n ollessa MQTT:ssä käytössä, sitä kutsutaan secure-mqtt:ksi, jonka portti 8883 nimenomaan on varattu MQTT:ssä TLS:n käyttöön [16 s. 22-23].

J. Cynthia et al. mukaan SSL:n toimintaperiaatteena on, että se käyttää kahta salausavainta tietojen salaamiseen ja näitä kahta avainta kutsutaan julkisiksi ja yksityisiksi avaimiksi [2 s. 20]. Yksityiset ja julkiset avaimet ovat pareiksi linkitettyjä tekstitiedostoja (käytännössä ne ovat bittijonoja), jotka muodostetaan omiksi pareiksi luomalla CSR (Certificate Signing Request) allekirjoituspyyntövarmenne. SSL.com mukaan yksityistä avainta käytetään sertifikaatti CSR:n digitaaliseen allekirjoittamiseen. Yksityinen avain toimii myöhemmin yhteyksien suojaajana ja palvelimen varmistamiseen. Mikäli kadottaa yksityisen avaimen tai se on mahdollisesti vaarantunut, tietoturvapalvelu SSL.com suosittelee luomaan varmenneavaimen uudelleen [17].

Mikäli tarvitaan uusi varmenneavain, on luotava ja lähetettävä uusi CSR allekirjoitusvarmennepyyntö. Yksi varmenteen myöntäjistä SSL.com tietoturvapalvelu voi myöntää uuden varmenteen, jolloin saadaan käyttöön uusi avainpari. Julkista avainta on levitettävä mahdollisimman laajasti. Julkiseen avaimeen sisältyy SSL-varmenne, jolla varmistetaan tiedon salaus, varmennus ja ettei tietoja käsitellä kuljetuksen aikana. Julkinen avain toimii yksityisen avaimen kanssa. Julkisella avaimella varmistetaan, että viesti on aito, eikä varmistamiseen tarvita salaista yksityistä avainta [17]. J. Cynthia et al. toteavat julkaisussaan, että jos verkkoselaimessa yritetään ottaa yhteyttä verkkosivustolle SSL:ää käyttämällä, selain pyytää verkkopalvelinta tunnistautumaan. Ohjelma kehottaa verkkopalvelinta

lähettämään selaimelle kopion SSL-sertifikaatista, jotta selain voi tarkistaa aitouden ja onko SSL-sertifikaatti luotettava. SSL-sertifikaatin ollessa luotettava, selain lähettää viestin web- palvelimelle. Istunto alkaa, kun palvelin kuittaa digitaalisella allekirjoituksella ja sen jälkeen voidaan jakaa salattuja tietoja palvelimen ja selaimen välillä. Man-in-the-middle -hyökkäykset ovat ongelmina SSL:ssä, koska se vaatii useampia lähetettäviä viestejä jokaista kättelyä kohden, ennen kuin tietojen siirtäminen suojatun tunnelin kautta olisi mahdollista [2 s. 20].

TLS protokolla tarjoaa yksityisyyttä ja tietojen eheyttä kahden kommunikoivan sovelluksen välillä, joten TLS on nykyään J. Cynthia et al. mukaan eniten käytetty suojausprotokolla. TLS:ää käytetään verkkoselaimissa ja muissa sovelluksissa, joissa vaaditaan tietojen turvallista vaihtoa verkon kautta, esim. tiedostojen siirrossa. J. Cynthia et al. mukaan TLS eroaa edeltäjästään SSL:stä siten, että TLS on protokollana turvallisempi ja tehokkaampi [2 s. 20]. Lisäksi TLS tukee uudempia ja turvallisempia algoritmeja viestien todennuksessa, avainmateriaalien luomisessa ja salaussarjan tukemisessa. TLS:n ominaisuutena on myös pitää yhteys aktiivisena, vaikka tietoja ei olisi lähetetty eli se on toimintavalmiudessa tietojen lähetysprosessin alkaessa. Vaikka TLS ja SSL eivät ole toisilleen yhteensopivia, TLS:ää voidaan kuitenkin käyttää joissain vanhoissa järjestelmissä [2 s. 20]. Kuitenkin A. Viswanathan toteaa, että TLS:ää käyttäessä sen mukana tulee lisäkuormaa. Mutta TLS:n suorituskykyä voi parantaa erilaisilla tekniikoilla, kuten istuntojen jatkamisella. Istunnon jatkaminen antaa asiakkaille mahdollisuuden käyttää valmiiksi neuvoteltua TLS-yhteyttä muodostamalla yhteyden uudestaan palvelimelle välttyäkseen täyden kättelyn aiheuttamaa ylikuormitusta [16 s. 22-23].



Kuva 4. Epäsymmetrinen ja symmetrinen kryptografiat[5]

3.2. DDS-protokolla ja havaitut haavoittuvuudet

DDS säätiön mukaan DDS (Data Distribution Service) protokolla integroi järjestelmät ja komponentit yhteen tarjoten matalan viiveen datayhteyttä, lisäksi se on arkkitehtuuriltaan erittäin luotettava ja skaalautuva. DDS säätiö julkaisun mukaan DDS on datakeskeisyydessään ainutlaatuinen ja se sopii ihanteellisesti IioT:hen, eli teolliseen esineiden Internetiin [18]. Myös P. Peniak ja M. Franekova ovat julkaisussaan ottaneet huomioon näkökohtia, joiden perusteella he ovat ehdottaneet parhaaksi ehdokkaaksi DDS protokollaa sulautetuissa järjestelmissä. Lisäksi DDS tarjoaa matalaa latenssia, sekä suuren suorituskyvyn tietoliikenteessä [19 s. 4]. Myös varsin laajat QoS-palvelulaatuparametrit ohjaavat tarkasti tietojen kulkemista väylän kautta julkaisijalta ja tilaajalta. Väliohjelmisto sovittaa julkaisijat tilaajiin tyypin, aiheen ja QoS tason perusteilla, jolloin tuloksena on nopea, suora ja kontrolloitu viestintä [19 s. 4].

P. Peniak ja M. Franekova toteavat julkaisussaan, että DDS-palvelu luo julkaisijan määrätylle toimialueelle. Luodulle julkaisijalle voi sovellus pyytää määrittämään DataWriter-objektin. Tietojen ollessa kirjoitettuina tieto-objektin tiettyihin aiheisiin, julkaisija tekee ilmoituksen uusista tiedoista muille solmuille. Lisäksi P. Peniak ja M. Franekova toteavat, että tietyt sovellukset vaativat tietojen

5 Kuvien alkuperäinen lähde https://www.tutorialsteacher.com/https/how-ssl-works.

lukemista DDS-protokollan kautta. Lisäksi tilaajille on luotava entiteetit toimialueille ja samassa yhteydessä on oltava myös liityntä DataReader-palveluun. DataReader-palvelun ollessa luotuna, voidaan aloittaa datan lukuprosessi. Kun tieto-objektille on määritetty aihetunnus, tilaaja lähettää tietoja tapahtumista DataReader-palvelulle. Tämän jälkeen kuuntelija saa ilmoituksen siitä, että tiedon lukeminen voidaan aloittaa [19 s. 3].

G. Nebbione & M. C. Calzarossa toteavat artikkelissaan, että IoT-ratkaisujen osalta DDS ei ole kaikkein tyypillisin protokolla, mutta DDS soveltuu joihinkin teollisuuden sovelluksiin, mm. lentoliikenteen ohjaukseen, älykkäisiin verkonhallintoihin, autonomisiin ajoneuvoihin jne. Lisäksi G. Nebbione & M. C. Calzarossa toteavat artikkelissaan DDS:n omaavan monipuolisen tietoturva mekanismin, lisäksi DDS tukee myös TLS-suojausmenetelmää. DDS protokolla täyttää hyvin vaatimukset muun muassa luottamuksellisuudesta ja eheydestä. Nämä ominaisuudet mahdollistavat uusimman OMG (Object Management Group) DDS-suojausspesifikaatio arkkitehtuurin, joka perustuu joukkoon sisäänrakennettuja laajennuksia [13 s. 9]. DDS sisältää suojausmekanismeja, joiden ominaisuuksiin kuuluu antaa todennuksia, pääsynhallintaa, luottamuksellisuutta, sekä eheyttä tiedon jakamiselle. Teollisen esineiden Internetin ympäristöjen tehtäväkriittisyyksien osalta on turvallisuuden kannalta tärkeää suojata IIoT-laitteiden ympäristöjä [18]. DDS Security käyttää niin sanottua hajautettua peer-to-peer -arkkitehtuuria, jonka ominaisuus on tarjota suojaa tinkimättä reaaliaikaisesta suorituskyvystä [18].

G. Nebbione & M. C. Calzarossa toteavat artikkelissa, että DDS:ssä haavoittuvuudet rikkovat luottamuksellisia yhteyksiä ja siten antavat mahdollisuuksia hyökkääjien kerätä dataa ja käyttää niitä vääriin tarkoituksiin. G. Nebbione & M. C. Calzarossa toteavat artikkelissaan myös, että ei ole olemassa lisäosia, jotka takaisivat täysin turvallisen DDS-ympäristön. Accel Control -laajennuksesta on löydetty joitakin vakavia haavoittuvuuksia [13 s. 11]. Muun muassa CVE-2019-15136 on haavoittuvuus, jossa DDS-osioiden etäosallistujayhteyksien käyttöoikeuksia ei tarkisteta, vaan haavoittuvuudella voidaan ohittaa tämä suoritus [20].

3.3. mDNS-palvelunetsintäprotokolla

G. Nebbione & M. C. Calzarossa toteavat julkaisussaan, että Multicast Domain Name System (mDNS) on avoin protokolla, jota käytetään laajasti palveluiden etsimiseen, sekä paikallisten linkkien nimien selvittämiseen. Yhdistettynä DNS-pohjaiseen (Domain Name System) palveluun, se tarjoaa joustavuutta ympäristössä, joissa on välttämätöntä yhdistää automaattisesti uusia laitteita [13 s.10]. Laitteet suorittavat kyselyjä lähialueella, eli paikallisella toimialueella monilähetyksen kautta. Lisäksi mDNS suorittaa kyselyjä DNS-palvelimiin unicastin avulla. Lisäksi mDNS lisää DNS määritykseen .local-toimialueeseen. B Villaverde et al. toteavat, että mDNS-mainoksilla voidaan tarjota tietoja palveluista ja resursseista niille laitteille, jotka ovat hajautetuissa DNS-pohjaisissa palveluhauissa [21 s. 10].

Ionos toteaa verkkojulkaisussaan, että jokainen istuntoon osallistuja saa tiedon nimen ja IP-osoitteiden välisestä yhteydestä ja täten voivat tehdä vastaavia merkintöjä mDNS-välimuistiin. Isäntänimeä ei tarvitse erikseen pyytää niin kauan kuin merkintä on voimassa. Multicast DNS luo paljon verkkoliikennettä, mutta kuitenkin se yrittää säästää aktiivisia verkkoresursseja. Tällä tarkoitetaan sitä, että asiakkaan tehdessä pyyntöä, asiakas lähettää sillä hetkellä olevan välimuistimerkinnän mukaisen vastauksen, joka on DNS:n mielestä oikea. Vastaanottajan tulee vastata vain silloin, kun vastaus ei ole enää oikein tai merkintä on ehtinyt vanhentua [22].

Vaikka mDNS:n käyttö on helppoa, niin sen käyttämiseen liittyy joitakin ongelmia, joista yksi on itse monilähetysprosessissa. Ionos verkkojulkaisun mukaan multicast DNS pyrkii pitämään verkkoliikenteen alhaisena, koska se vaatii tietokoneilta jatkuvaa verkkovalvontaa ja saapuvien viestien käsittelyä. Jatkuva verkkovalvonta ja viestien käsittely rasittaa prosessointitehoa. Ongelmana on myös isäntänimien allokointi, joka tarkoittaa, että jokaiselle laitteelle voidaan antaa vapaasti nimi, mikäli se päättyy .local-päätteeseen. Teoriassa se voi johtaa siihen, että kahdella osallistujalla voi olla verkossa sama isäntänimi ja tätä ongelmaa eivät ole mDNS:n kehittäjät ottaneet ratkaistavaksi tietoisesti [22]. G. Nebbione & M. C. Calzarossa ovat todenneet mDNS-protokollan heikon tietoturvan, jonka vuoksi mDNS:ään olisi tärkeää kehittää parempia suojauksia. Protokollan turvallisuusuhista G. Nebbione & M. C. Calzarossa ovat tehneet analyyseja mahdollisista turvallisuusuhista tunnisteita ja luokituksia hyökkäyksistä [13 s. 11].

Ensimmäisenä turvallisuusuhkana G. Nebbione & M. C. Calzarossa mainitsevat palvelunestohyökkäykset, joissa hyökkääjät iskevät viesteillä mDNS-yhteensopivuussolmuihin ja käyttävät hyväksi protokollan tiettyjä ominaisuuksia, kuten niin sanotuilla häiriöviesteillä pyritään saamaan solmuihin virhetilan, eli solmut eivät pysty vastaamaan pyyntöihin, tai viesteillä pyritään mitätöimään välimuistimerkintöjä tai estämään tutkittavia prosesseja [13 s. 11]. Toinen haavoittuvuus on myrkytyshyökkäys, jossa hyökkääjät huijaavat mDNS-vastausviestejä ja samalla mainostavat palveluväärennöksiä, joita sitten käytetään hyökkäyksissä tuntemattomien solmujen kimppuun. Kolmas haavoittuvuustyyppi on etähyökkäykset. Etähyökkäyksessä hyödynnetään mDNS-yhteensopivia solmuja. Solmukohdat vastaavat ulkopuolella tuleviin väärinkäyttöpalvelu kyselyihin. Tämä antaa hyökkääjille mahdollisuuden suorittaa hajautettuja palvelunestohyökkäyksiä, joiden tavoite on kerätä arkaluontoisia tietoja [13 s. 11].

G. Nebbione & M. C. Calzarossa ovat suorittaneet testianalyyseja 29 CVE:lle, joilla on haavoittuvuuksia mDNS-pohjaisissa tuotteissa ja palveluissa. Analyysit osoittivat mDNS:ssä olevan solmukohtia, jotka olivat tahattomasti vastanneet unicast-kyselyihin, kyselyt olivat paikallisten linkkien ulkopuolella olevia lähdeosoitteita. CVE-2017-6520 on mDNS:ssä esiintyvä haavoittuvuus, jonka tarkoituksena on saada mDNS-vastaaja vastaamaan IPv4-onecast kyselyjä lähdeosoitteisiin, jotka eivät ole paikallisesti linkittyneitä. Toimenpiteen ansiosta etähyökkääjät voivat aiheuttaa palvelunestoja tai kerätä mahdollisia arkaluontoisia

tietoja UDP-paketeista portin 5353 kautta. Lisäksi on mahdollista suorittaa palvelunestohyökkäys lähettämällä väärin muotoiltuja ja haitallisesti muodostettuja paketteja [23] [13 s. 11].

3.4. SSDP-verkkoprotokolla ja protokollan haavoittuvuudet

SSDP eli Simple Service Discovery Protocol on pienissä verkoissa käytettävä verkkoprotokolla, jota käytetään muun muassa kotiverkoissa. Lisäksi SSDP:tä käytetään verkkomainostuksissa, sekä löytämään UPnP (Universal Plug-and-Play) -arkkitehtuuria tukevia verkkopalveluja [24]. Lisäksi kyberturvallisuuskeskus StormWall mainitsee SSDP:n olevan UPnP-arkkitehtuurille merkittävä protokolla, koska sen avulla kodin laitteiden verkkoyhteydet toimivat samassa pienessä verkossa tai ovat yhteydessä samaan WiFi-pisteeseen. Laitteiden toimivuuden takaamiseksi SSDP:n tulee olla yhteensopivia UPnP-arkkitehtuurin kanssa [24].

G. Nebbione & M. C. Calzarossa mainitsevat tutkimuksessaan, että SSDP-yhteensopivien laitteiden tietoturvariskien vuoksi niihin kohdistuu monia hyökkäyksiä, kohteina ovat mm. palvelunhakuominaisuudet. SSDP-solmujen suurimpana uhkana voidaan pitää DdoS (Distributed Denial of Service) hyökkäyksiä, koska niiden on tarkoitus saada laitteet reagoimattomaan tilaan ja siten estämään palveluiden toimivuutta [13 s. 12].

DdoS-hyökkäyksissä hyökkääjät pyrkivät käyttämään verkkokerrokseen upotetun SSDP-protokollan haavoittuvuuksia. Verkkokerrokseen upotetulla SSDP:llä ilmeisesti kehittäjät ovat halunneet tehdä laitteelle mahdollisimman paljon yksinkertaisia vuorovaikutusmahdollisuuksia pienessä verkossa, joka on johtanut turvallisuuden heikentymiseen [24]. Lisäksi hyväksikäyttöä on edesauttanut UPD (User Datagram Protocol) ja SSDP-protokollalaitteiden virheelliset kokoonpanot, sekä edellä mainittujen protokollien ominaisuudet [13 s. 12]. G. Nebbione & M. C. Calzarossa toteavat, että SSDP-protokollalaitteissa esiintyvien virheellisten kokoonpanojen hyväksikäytössä hyökkääjä luo M-SEARCH-sanoman, jossa käytetään huijaus IP-osoitetta solmussa. Kyseinen viesti lähetetään haavoittuville SSDP- laitteille, jonka jälkeen solmukohteet täyttyvät hyökkäyksen vastausviesteistä ja niillä on iso vahvistuspotentiaali [13 s. 12]. Lisäksi Nebbione & M. C. Calzarossa toteavat, että SSDP-yhteensopivia solmuja ei uhkaa pelkästään edellä mainittu tietoturvauhka, myös passiiviset hyökkäykset ovat yhtä vaarallisia. Passiivisiin hyökkäyksiin lukeutuu mm. multicast- viestien salakuuntelu verkossa. Tämä tietoturvauhka on varsin ikävä, koska se ei erikseen varoita olemassaoloaan, vaan salakavalasti hyökkääjä voi saada arkaluontoisia tietoja ja tästä voi seurata vakava takaisku yksityisyyden ja luottamuksellisuuden kohdalla [13 s. 12].

SSDP:hen kohdistuvia hyökkäyksiä voidaan kuitenkin minimoida, vaikka niitä ei voi saada kokonaan pois. StormWall kyberturvallisuuskeskus on antanut ohjeeksi mm. estää UDP-portti 1900:lle saapuvaa ja lähtevää liikennettä, kyseinen esto asetetaan palomuurissa. Lisäksi on hyvä käyttää BGP (Border Gateway

Protocol) flowspec- toimintoa, jolla rajoitetaan tulevaa liikennettä porttiin 1900. StormWallin mukaan kannattaa käyttää myös UDP-pohjaisia palveluita varauksella, koska UDP:n kohdistuvia DdoS- hyökkäyksiä on varsin vaikeaa torjua. Lopuksi kyberturvallisuuskeskus StormWall kehottaa tarkistamaan, mitkä verkkoon kytketyistä laitteista ovat tarpeettomia, sitten tarpeettomat SSDP-toiminnot voidaan kytkeä pois päältä [24].

Protokolla	Viestit	Hajautettu	Julkaisija/ lähettäjä	Pyyntö/ vastaa	Salauspalvelu	TCP	UDP	TLS	DTLS
MQTT	Kyllä	Ei	Kyllä	Ei	Kyllä	Kyllä	Ei	Kyllä	Ei
DDS	Kyllä	Kyllä	Kyllä	Osittain	Kyllä	Kyllä	Kyllä	Kyllä	Kyllä
mDNS	Ei	Kyllä	Ei	Kyllä	Ei	Ei	Kyllä	Ei	Ei
SSDP	Ei	Ei	Ei	Kyllä	Ei	Ei	Kyllä	Ei	Ei

Taulukko 2. Vertailussa neljä IoT-protokollaa.[6]

3.5. MQTT-tiedonsiirtoprotokolla

MQTT on tiedonsiirtoprotokolla, joka kuuluu avoimen standardin protokolliin. G. Nebbione & M. C. Calzarossa toteavat, että MQTT on tällä hetkellä laajasti käytetty protokolla IoT-kontekstissa, koska kyseinen protokolla on selvästi monipuolinen, kevyt ja on siten IoT-skenaarioihin ihanteellinen. MQTT sopii hyvin tilanteisiin, kun halutaan säästää verkkokaistan leveyttä ja laskentatehoa [13 s. 5]. Mosquitto Eclipsen verkkosivustolla on tarkkoja ohjeita välittäjäohjelman konfigurointiin. Protokollassa ei ole tarvetta tehdä konfigurointia, pelkkä julkaiseminen riittää perustoimintojen suorittamiseen. Mutta haluttaessa lisätoimintoja ja parannettaessa tietoturvaan liittyviä toimintoja, on lisättävä konfigurointiin uusia toimintoja [25].

Eclipse-säätiön mukaan yksi MQTT-protokollan ominaisuus on protokollan viestien aiheiden julkaisut. Viestien aiheita käsitellään hierarkioina käyttämällä kauttaviivaa erottimena. Ominaisuus antaa mahdollisuuden luoda yhteisiä teemoja järkevässä järjestyksessä samalla tavalla kuin tiedostojärjestelmissä [25]. MQTT-protokollan ominaisuuksiin kuuluu viestien säilyttäminen tietokantatiedostossa. Mosquitto luo .db-tiedoston, jos toiminto on määritelty konfiguraatiotiedostossa. Jos välittäjä ei saa yhteyttä tilaajaan, tulee viestit säilyttää. Mikäli tulee uusi tilaus, jonka aihe on sama kuin säilytetyn viestin aihe, tilaaja saa viestin [6].

6 Alkuperäinen taulukko osoitteessa http://peg.unipv.it/publications/PDF/future-internet_preprint.pdf.

3.5.1. MQTT:n turvallisuusongelmat ja toimenpiteet

G. Nebbione & M. C. Calzarossa ovat tehneet omia analyyseja ja ovat tulleet siihen tulokseen, että MQTT:stä on tunnistettu useita haavoittuvuuksia ja he ovat luokitelleet haavoittuvuuksia prosesseissa. Todennuksessa MQTT-välittäjä ei ole tarkistanut julkaisijan/tilaajan identiteettejä kunnolla, eikä kykene estämään todennusyritysten kertoja, jolloin hyökkääjä voi toistaa todennusyrityksiä niin monta kertaa, kunnes hyökkääjä pääsee kirjautumaan sisään. Mikäli hyökkääjä pääsee läpi todennuksesta, hyökkääjä voi häiritä välittäjää ylikuormittamalla sitä. Pahimmassa tapauksessa hyökkääjä saisi kaadettua MQTT-välittäjän ylikuormituksen seurauksena [13 s. 5]. Toinen esimerkki haavoittuvuudesta on valtuutuksessa, jossa välittäjä ei voi asettaa julkaisu/tilausoikeuksia oikein, tällöin hyökkääjällä olisi mahdollisuus kontrolloida MQTT-laitteissa sisältäviä tietoja ja toimintoja [13 s. 5].

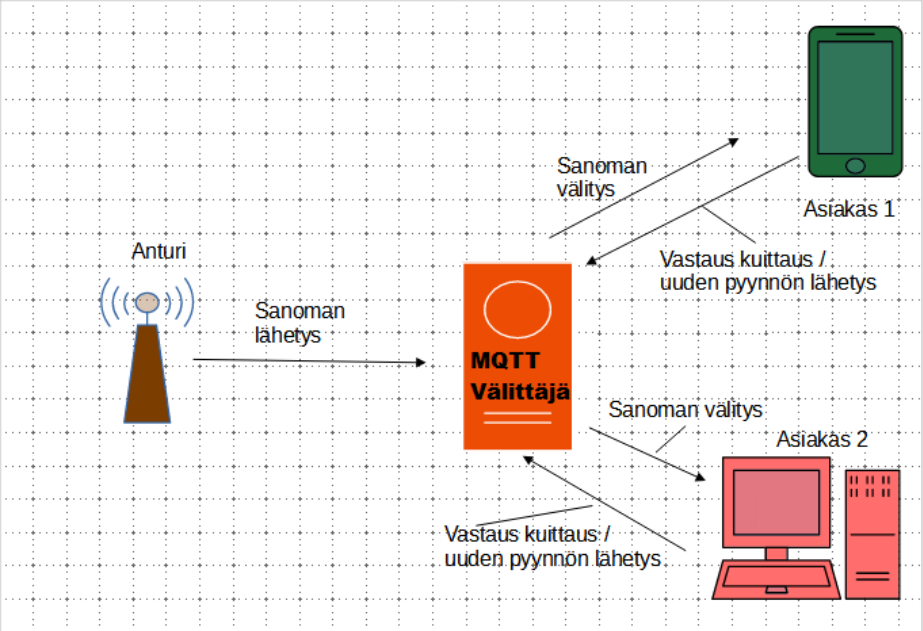
A. Viswanathan käsittelee julkaisussaan todennusmekanismia, joka on tarjolla sovellustasolla MQTT-protokollassa. Mekanismin avulla asiakas voi todentaa itsensä käyttäjätunnuksen ja salasanan avulla. Asiakkaan muodostaessa ensimmäisen yhteyden välittäjään, on asiakkaalla mahdollista lähettää käyttäjätunnus ja salasana CONNECT-pyynnön kanssa. Mikäli välittäjä kieltää anonyymisiä yhteyksiä ja pitää yllä salasanalistaa, jonka avulla se ristiviittaa asiakkaan yrittäessä muodostaa yhteyden, on asiakkaan annettava oikea käyttäjätunnus ja salasana, muussa tapauksessa yhteys katkeaa ja asiakas ei pääse muodostaa yhteyttä välittäjään [16 s. 23].

Turvatoimenpiteinä Wallarm tarjoaa monikerroksista API-turvaratkaisua ja sitä voidaan käyttää erilaisissa pilviympäristöissä. Wallarm API-tietoturvalla on käytössä vahvat API-testauskäytännöt. Testauskäytännöt auttavat kehittäjää havaitsemaan API-tietoturva-aukkoja jo niiden alkuvaiheessa ja minimoimaan vahingot mahdollisimman vähiin. MQTT-palvelimien tietoturvan osalta kulunvalvonta, käyttäjätunnus ja salasanasuojaus ovat riittäviä. Lisäksi tulee ottaa huomioon MQTT:n toimivuus TCP/IP:ssä, jolloin MQTT-suojaus on vasta puoliksi valmis. Suojausongelman ratkaisuna on SSL/TLS-toteutus, joka ratkaisee koko ongelman, mutta samalla se tekee kevyestä viestinnästä vähän raskaamman [26]. Myös G. Nebbionen & M. C. Calzarossan mukaan MQTT käyttää salaustekniikassa pohjana TLS-salausta, sekä muita SASL (Simple Authentication and Security Layer) standardin mukaisia todennusmekanismeja [13 s. 4]. G. Nebbione & M. C. Calzarossa toteavat myös, että kyseiset palvelut eivät riitä MQTT-laitteiden suojauksessa. DEFCON 24:n MQTT-standardiraporteissa on osoitettu, että välittäjien virheellisestä konfiguraatiosta on koitunut tietoturvariskejä, sekä useita ohjelmistoissa ilmenneitä haavoittuvuuksia. Tämä taas lisää järjestelmän/palvelun hyväksikäytön riskejä [13 s. 5].

MQTT:ssä on myös ongelmana viestien näkyvyys. Viestien lähtiessä MQTT:ltä käyttäjätunnukset ja salasanat ovat helposti saatavilla, koska ne ovat sisällöltään selkokielistä tekstiä. Todennusprosessin tukemisessa käytetään TLS-

suojausta. TLS-suojauksen avulla viestit saadaan hyvin piilotettua. Suojauksella halutaan suojata arkaluontoisia tietoja mm. käyttäjätunnuksia, salasanoja ja muita tunnistetietoja [2 s. 24-25].

MQTT on protokollaltaan kevytrakenteinen, mutta protokollalla on joitakin haittapuolia liittyen rajoitettuihin laitteisiin. MQTT-asiakkaiden on tuettava TCP:tä ja pidettävä aina yhteys auki välittäjään. Lisäksi joissakin ympäristöissä voi esiintyä suuria pakettihäviöitä tai laskentaresursseissa on niukkuutta. Lisäksi MQTT-aiheet (topic) ovat usein pitkiä merkkijonoja. Puutteet ovat kuitenkin korjattu MQTT-SN protokollalla (MQTT-Sensor Networks), jolla määritellään MQTT:n UDP-kartoitukset ja sillä lisätään välittäjälle tuki nimiaiheiden indeksointiin [2 s. 24-25].



Kuva 5. Kuvassa on esitetty anturilta lähtevä sanoma MQTT-välittäjälle. Välittäjältä lähtee sanomat asiakaslaitteille ja sen jälkeen ne voivat lähettää uuden pyynnön välittäjälle, jotta saavat päivitettyjä sanomia.[7]

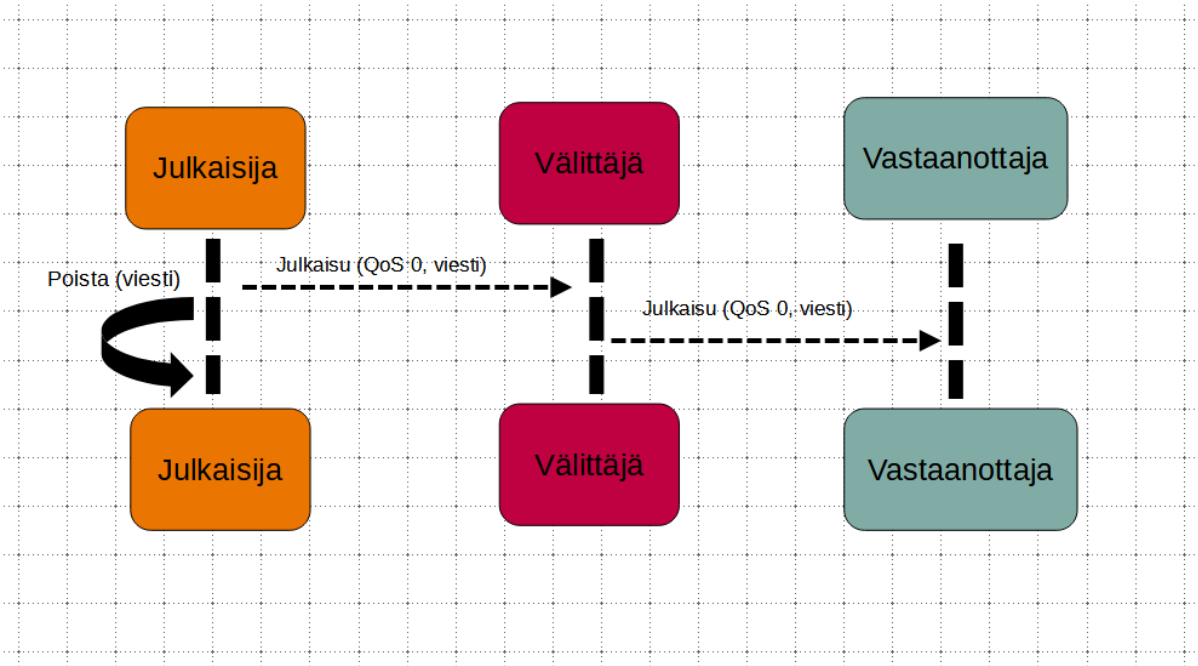
3.5.2. Viestien lähetykset QoS-palvelutasoilla

MQTT-protokollassa yhtenä ominaisuutena on palvelulaatujen määrittelyt, jotka takaavat luotettavan viestien välityksen erilaisille verkkoympäristöille. Tätä varten MQTT:lle on kehitetty omaksi ominaisuudeksi QoS (Quality of Service), joka on suunniteltu MQTT-protokollan IoT-skenaarioita varten. MQTT:n toimintaskenaariot eivät ole suunnattu pelkästään PC:lle, vaan niitä käytetään myös laajoissa kapeakaistaisissa verkoissa. Toimintaskenaarioita on myös pienitehoisissa laitteissa. Mikäli laatuongelmat lähetyksissä pystytään ratkaisemaan protokollakerroksessa, silloin tämä tarjoaa suurta joustavuutta IoT-sovellusten kehittämisessä [27].

7 Alkuperäinen kuva osoitteessa https://mqtt.org/.

A. Viswanathan käsittelee julkaisussaan QoS-palvelun laatutasoja ja todennut QoS:n olevan MQTT:lle tärkeä ominaisuus. MQTT:n QoS-ominaisuudet soveltuvat yksinkertaiseen viestintään epäluotettavissa verkoissa. Lisäksi MQTT-protokolla vastaa uudelleenlähetyksen käsittelyistä ja takaa viestin toimittamisen taustalla olevaan kuljetuskerrokseen, joten asiakkaat voivat valita QoS:n tason ja kyseisen tason verkkoinfrastruktuurin mukaisesti [16 s. 18]. F-Securen mukaan on huomioitava, että MQTT-julkaisuviestien QoS-tasot eivät välttämättä säily samana lähettäjältä asiakkaalle. QoS-tasot voidaan määritellä uudelleen asiakkaan ja palvelimen välillä [9]. QoS-tason valinnassa otetaan huomioon vielä se, että mitä korkeampi QoS-taso on, sen monimutkaisempia prosesseja se pystyy käsittelemään, mutta samalla se kuluttaa enemmän järjestelmäresursseja [27].

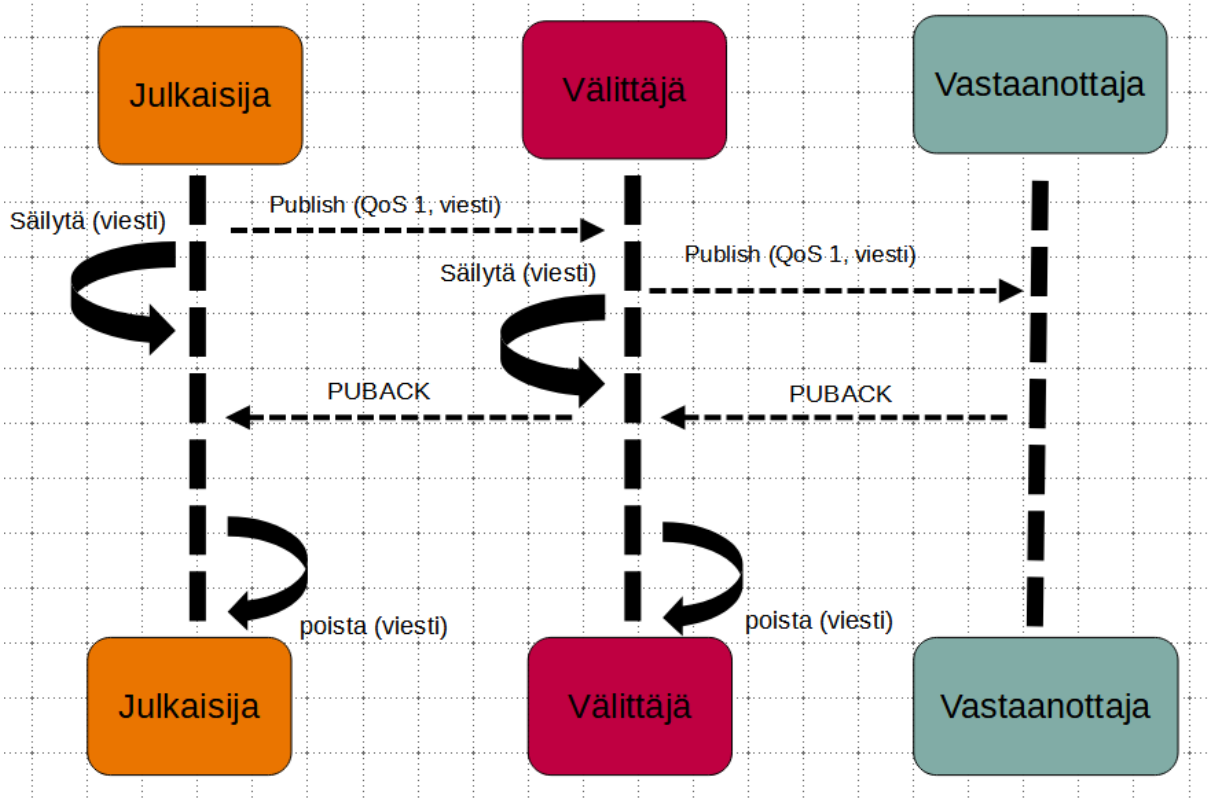
QoS-tason määrityksessä kerrotaan, että millä tasolla viestin välittäjät/asiakkaat vastaanottavat viestejä. Viestit voidaan lähettää kaikilta QoS-tasoilta. Asiakkaat voivat tilata aiheita kaikilla tasoilta. Esimerkiksi, jos viesti julkaistaan QoS-taso 2:ssa ja asiakas on tilannut viestin tasolla QoS 0, silloin viesti lähetetään tasolla QoS 0. Mikäli samalla viestillä on toinenkin asiakas, mutta kyseinen asiakas tilaa viestin tasolla QoS 2, asiakas saa tismalleen saman viestin, kuin toinen asiakas, joka tilasi viestin tasolla QoS 0. On kuitenkin huomioitava, että korkeimmat QoS-palvelutasot ovat luotettavampia, mutta ne taas verottavat kaistanleveyttä ja niissä esiintyy suurempi latenssi [27]. A. Viswanathan toteaa julkaisussaan QoS 0 -tason sisältävän vähiten lisäkustannuksia aiheuttavia tekijöitä. A. Viswanathan toteaa, että QoS 0-tasolla vastaanottaja ei kuittaa viestejä, eikä tallenna viestejä tai lähetä viestejä uudestaan. QoS 0-taso tarjoaa samat takuut kuin sen alla oleva TCP-protokolla [16 s. 18]. Lisäksi A. Viswanathan toteaa julkaisussaan, että QoS 0-tasolla on vähiten lähetyksiä, koska se sisältää fire & forget, eli lähetä ja unohda ominaisuuden. Siitä huolimatta, että taustalla toimii TCP/IP, viestien välittämiseksi ei suoriteta ylimääräisiä ponnisteluja. Viestin lähetyksessä lähetetään PUBLISH-viesti. Mikäli viesti häviää verkossa, sitä ei voida enää palauttaa [16 s. 33].



Kuva 6. QoS 0-palvelutasolla viestin julkaisu ja vastaanotto välittäjän kautta.[8]

A. Viswanathan käsittelee julkaisussaan QoS 1 -palvelun laatutason varmistavan viestin toimittamisen perille vähintään kerran. Mikäli lähettäjä ei saa kuittausta vastaanotetusta, lähettäjä asettaa DUP-lipun eli kaksoiskappaleen ja lähettää viestin uudelleen niin kauan, kunnes on saanut kuittauksen viestistä. Lisäksi A. Viswanathan toteaa, että QoS 1 -palvelutason ollessa käytössä tallennetaan viesti paikallisesti siltä varalta, että lähetyksessä tapahtuu joku odottamaton virhe. Sen jälkeen, kun lähettäjä on saanut kuittauksen viestistä, viesti poistetaan [16 s. 18]. Kuvassa 7 julkaisija lähettää QoS 1 -palvelutasolla viestin ja odottaa PUBACK-vastausta.

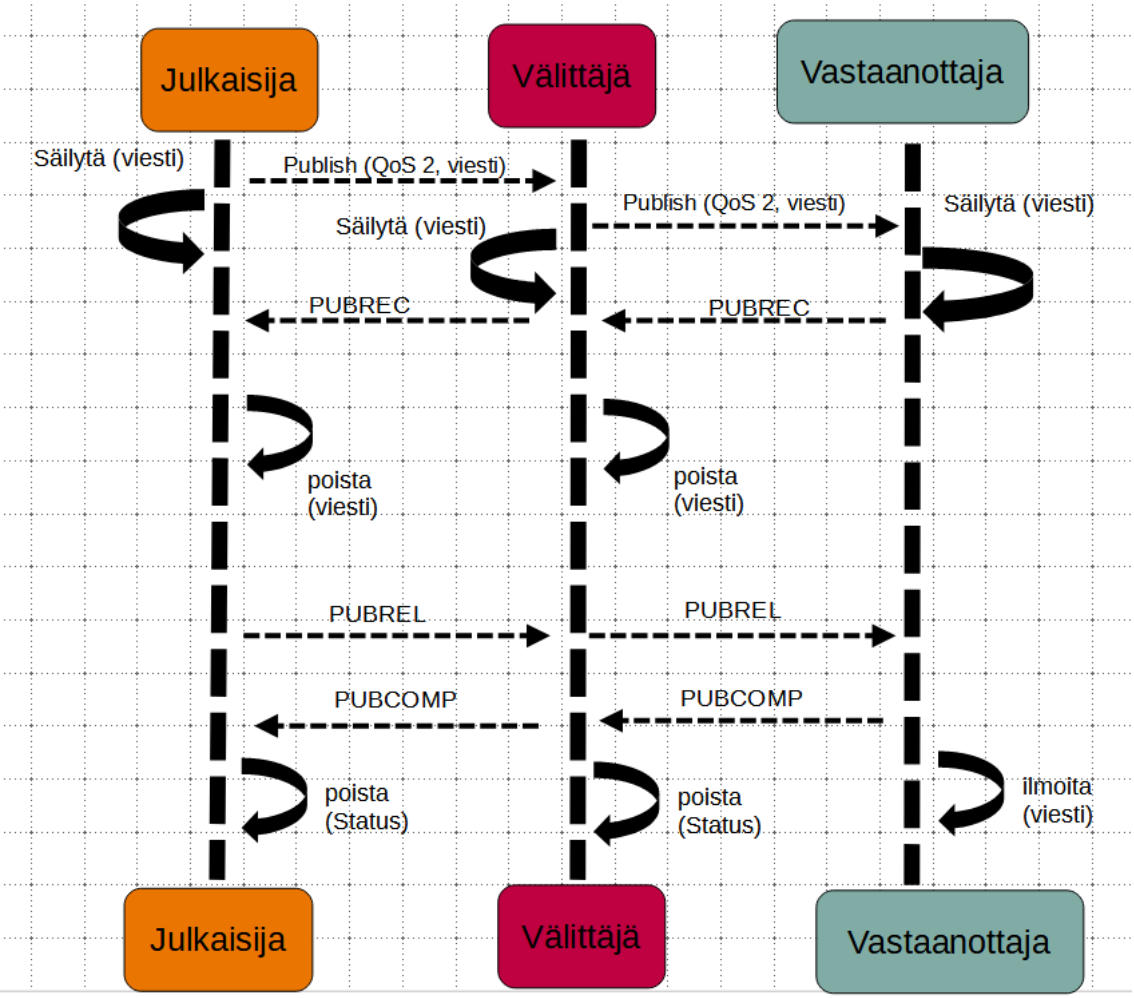
8 Alkuperäinen kuva osoitteessa https://www.emqx.com/en/blog/introduction-to-mqtt-qos.



Kuva 7. QoS 1 -palvelutasolta viesti menee välittäjälle ja siitä vastaanottajalle.[9]

A. Viswanathan käsittelee julkaisussaan QoS 2 -palvelun laatutason olevan korkein MQTT-protokollan tarjoama QoS-taso. QoS 2 -tasolla taataan, että vastaanottaja saa varmasti vastaanotettua jokaisen viestin tasan kerran. Tasoltaan QoS 2 on hitain, mutta se on tasoltaan kaikkein luotettavin. Taso on käytössä, jos lähetettävät viestit sovelluksesta ovat kriittisiä ja niiden on päästävä varmasti perille. Viestien vastaanottaminen on varmistettava kuittausviesteillä [16 s. 18-19]. Kuvassa 8 esiintyy palvelutaso QoS 2, joka on palvelulaadun tasolta korkein. Molemmat, sekä julkaisijat, että tilaajat varmistavat viestin julkaisun vain kerran kahden istunnon aikana.

9 Alkuperäinen kuva osoitteessa https://www.emqx.com/en/blog/introduction-to-mqtt-qos.



Kuva 8. QoS 2 -palvelutaso viesti julkaisijalta välittäjälle ja välittäjältä vastaanottajalle.[10]

MQTT sanomat	Tehtävä
CONNECT	Muodostaa yhteyden välittäjään
CONNACK	Välittäjän vastaus julkaisijalle
PUBLISH	Kertoo olevansa julkaisija
SUBSCRIBE	Kertoo olevansa tilaaja
PUBREL	Välitysviesti tilaajille
PUBREC	Kuittausviesti. Välittäjä saanut asiakkaan lähettämän viestin (QoS 2 -tason viestintä)
PUBCOMP	Kuittausviesti onnistuneesta vastaanotetusta viestistä (QoS 2 -tason viestintä)
PUBACK	Kuittaa viestin vastaanottamisen (QoS 1 -tason viestintä)

Taulukko 3. MQTT-sanomat ja niiden tehtävät.

10 Alkuperäinen kuva osoitteessa https://www.emqx.com/en/blog/introduction-to-mqtt-qos.

3.6. Eclipse Mosquitto-välittäjäohjelma

Eclipse säätiön verkkosivustolla on selostettu tarkat tiedot Eclipse Mosquitto avoimen lähdekoodiviestivälittäjän ohjelmasta, jolla on EPL (Eclipse Public License) lisenssi. Mosquitto ohjelma toimii useissa MQTT-protokollaversioissa 3.1:stä 5.0:n. Mosquitto ohjelma on kevyt ja ohjelmaa on saatavilla useille alustoille (Windows, Linux, Mac), sekä pienitehoisille tietokonelaitteille, pienistä mikrokontrollereista (esim. Raspberry Pi) aina palvelinkoneisiin. Mosquitto ja MQTT-protokolla toimivat yhdessä hyvin, koska MQTT-protokolla tarjoaa kevyitä menetelmiä viestien ajamiseen julkaisu-/tilausmalleissa [28].

Mosquitto ohjelma tarjoaa projekteihin C- kirjastoa nimenomaan MQTT-asiakkaan toteutuksissa. Lisäksi se tarjoaa erittäin suositut mosquitto_pub- ja mosquitto_sub-komentorivit MQTT-asiakkaille. Mosquittolla voidaan luoda muutamissa minuutissa omia pieniä testausprojekteja, lisäksi Mosquitto-Project:lla voidaan suorittaa asiakastestauksia erilaisilla menetelmillä testipalvelimen test.mosquitto.org -osoitteessa. Testejä voidaan suorittaan pelkästään MQTT:llä tai liittämällä testiin mukaan TLS-suojaus. Lisäksi voidaan testata MQTT WebSocketsia, sekä TLS:n kanssa, että ilman TLS:ää [28].

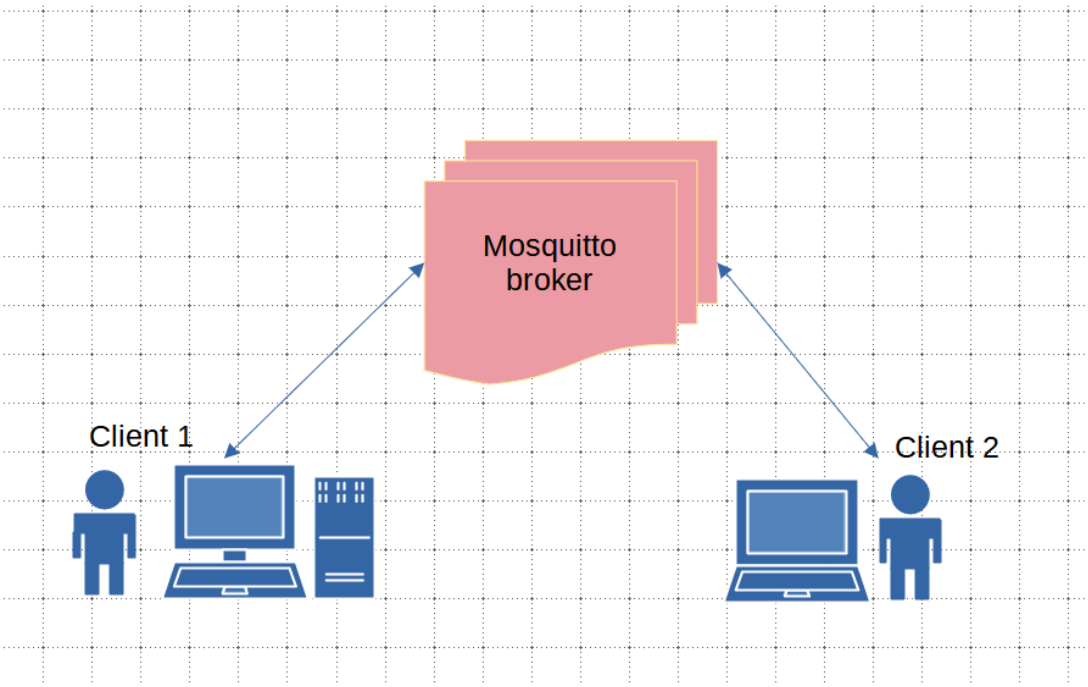
4. MQTT OMINAISUUKSIEN TODENTAMINEN

Työtä varten laadittiin seuraavat tutkimuskysymykset:

- Miten palvelutaso näkyy viestiliikenteessä?
- Onko viestin pysyvä tallennus tietoturvauhka?
- Miten MQTT:ssä asiakkaan salasana ja viesti salataan?
- Miten viestin kohdentaminen halutulle asiakkaalle parantaa tietoturvaa?

Teknisessä osuudessa on toteutettu MQTT-protokollan ominaisuuksien testaamista ja tavoite oli todentaa protokollan ominaisuuksien toimivuudet ja mahdolliset haavoittuvuudet. Ensimmäisessä testauksessa testattiin viestiliikenteen kulkua asiakkaan ja välittäjän kesken. On tärkeää ymmärtää, millaisia viestejä tiedonsiirtotapahtuma sisältää. Testeissä tarkasteltiin eri palvelutasojen ilmentymää viestiliikenteessä.

Testeillä selvitettiin voidaanko lähettää viesti kohdennetulle asiakkaalle, sekä salasanan kryptausta, koska molemmat ovat tietoturvan kannalta tärkeitä. Lisäksi testattiin viestiliikenteen salaamista käyttämällä MQTT:n mukaista TLS:lle standardoitua porttia 8883. Kappaleessa 3.5.1 osoitettiin G. Nebbione & M. C. Calzarossa, sekä A. Viswanathan esittämän kirjautumisyritysten rajoittamattomuus todeksi. Viimeisessä testauksessa testattiin tiedon pysyvää tallennusta, joka on tärkeää siinä vaiheessa, jos välittäjän ja vastaanottaja-asiakkaan välinen yhteys on poikki.



Kuva 9. Asiakas 1 lähettää viestin välittäjälle ja asiakas 2 vastaanottaa viestin välittäjältä.

4.1. Protokollatestauksissa käytetty laite ja järjestelmä

Tietokoneelle asennettiin välittäjäksi Mosquitto Broker, versio 2.0.14. Viestiliikenteen ohjaamisessa käytetään pääasiassa komentokehotteita. Konfiguroinnit suoritettiin Notepad++ tekstieditorilla. Muita hyödynnettyjä ohjelmia olivat mm. Wireshark versio 3.6.3, jolla voidaan seurata lähiverkkoliikennettä, WLAN-liikennettä, Ethernet-liikennettä ja silmukkaliikettä. Viimeksi mainitulla toiminnolla seurattiin Mosquitton liikennettä ja valitsemalla sopivia suodattimia, jolloin näytölle saatiin halutun protokollan liikenteet, esimerkkinä kappaleessa 4.5.1 kuva 25. Testaus-/mittauskertoja suoritettiin 3-5 kertaa, joilla pyrittiin varmistamaan mittauksien tuloksien oikeellisuudet. Viestien salaamista varten koneelle asennettiin OpenSSL v1.1.1m Light ja Mosquitton konfiguroinnissa TLS-versioksi määritettiin tlsv1.2. Alla olevassa taulukossa on kerrottu tietokoneen ominaisuuksia, joita on käytetty testien suorittamisessa.

Tietokone	Asus Tuf Gaming -kannettava tietokone
Käyttöjärjestelmä	Windows 10 Home -versio
Järjestelmätyyppi	64-bittinen
Suoritin	AMD Ryzen 5 2.1GHz kellotaajuus
Näytönohjain	Nvidia GeForce GTX 1050

Taulukko 4. Tietokoneen ominaisuudet listattuna.

4.2. Viestin lähetys ja viestiliikenteen mittaus

Aluksi suoritettiin kaksiosainen MQTT-protokollan ominaisuuksien testaus. Ensimmäisessä osassa testattiin viestien lähettämistä välittäjälle ja vastaanottamalla välittäjältä viestiä käyttäen kolmea eri palvelutasoa. Palvelutasoista tutkittiin, millä tavalla ne eroavat toisistaan lähetyksen aikana. Toisessa osassa mitattiin verkkoliikenteen kulkua kolmen eri palvelutason välillä ja lopuksi laadittiin vertailutaulukko siitä, miten verkkoliikenteen määrä vaihtelee eri palvelutasojen kohdalla. Verkkoliikenteen mittauksen toteutuksessa käytettiin sitä varten asennettua Wireshark tietokoneohjelmaa, joka tarkkailee verkkoliikennettä.

4.2.1. Viestin lähetys QoS 0-palvelutasolta

Kuva 9:n ylemmässä osassa havaitaan, että asiakas lähettää CONNECT-sanoman välittäjälle. Välittäjällä on velvollisuus vastata sanomaan CONNACK-viestillä, jos yhteys toimii. Tämän jälkeen julkaisija lähettää QoS 0-palvelutasoa käyttämällä julkaisijan sanoman välittäjälle. Kappaleessa 3.5.2 mainitut CONNECT ja

CONNACK ovat MQTT-protokollan ominaisuuksia, joilla varmistetaan yhteyden muodostuminen asiakkaan ja välittäjän kesken. Muita ominaisuuksia ovat mm. PUBLISH, jolla asiakas voi julkaista viestejä heti muodostettuaan yhteyden välittäjään. On tärkeää, että viesteissä on aihe (topic), joiden avulla välittäjä saa lähetettyä viestin aiheesta kiinnostuneille asiakkaille. DISCONNECT sulkee asiakkaan yhteyden välittäjään.

Kuvan 10 alaosassa vastaanottaja lähettää yhteyspyynnön ja sen jälkeen vastaanottaa viestin välittäjältä yhteyden onnistumisesta. Seuraavaksi asiakas lähettää SUBSCRIBE-viestin, joka sisältää viestin aiheen ja pyynnön viestin palvelutasoksi QoS 0. Viestien julkaisemista varten, asiakas lähettää SUBSCRIBE-viestin vastaanottaakseen asiakasta kiinnostavia viestejä. Vastaanottaja ei lähetä kuittausta viestistä, koska QoS-palvelutaso on 0. Tämä on osoitettu kappaleessa 3.5.2 kuvassa 6, jossa on esitetty QoS 0-palvelutason tapahtumat. SUBSCRIBE-ominaisuus mahdollistaa asiakasta tilaamaan viestejä.

SUBSCRIBE:lle on olemassa omat paluukoodeiksi kutsutut palautuskoodit, jotka ovat käytössä QoS-palvelutason pareina. Eri luvuilla on erilaiset toimintamerkitykset. Taulukko 5:ssa on esitetty palvelutasojen paluukoodit ja niiden merkitys. Lisäksi SUBACK on MQTT:n viesti, joka toimii kuittausviestinä, jonka välittäjä lähettää asiakkaalle. Vastaanottaja saa PUBLISH-viestin, joka sisältää julkaisijan viestin.

Paluu koodi	Paluu koodin vastaus
0	Onnistunut - maksimi QoS 0
1	Onnistunut - maksimi QoS 1
2	Onnistunut - maksimi QoS 2
128	Epäonnistunut

Taulukko 5. Palvelutason lähetyksien paluukoodit ja niiden merkitys.[11]

11 Taulukon alkuperäinen lähde https://www.hivemq.com/blog/mqtt-essentials-part-3-client-broker-connection-establishment/.

```
C:\mosquitto>mosquitto_pub -h localhost -t topic/pub -m "QoS 0 viestin toimitus" -r -i client -d -q 0
Client client sending CONNECT
Client client received CONNACK (0)
Client client sending PUBLISH (d0, q0, r1, m1, 'topic/pub', ... (22 bytes))
Client client sending DISCONNECT

C:\mosquitto>
```

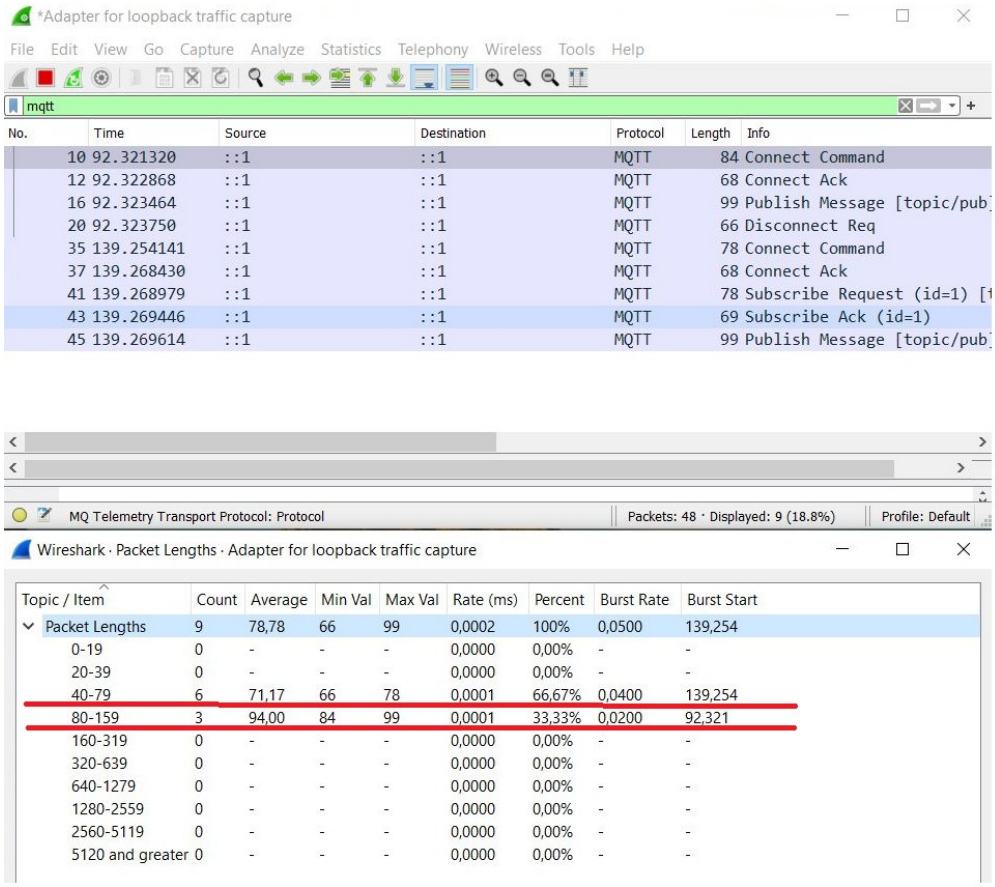
```
C:\mosquitto>mosquitto_sub -h localhost -t topic/# -d -q 0
Client null sending CONNECT
Client null received CONNACK (0)
Client null sending SUBSCRIBE (Mid: 1, Topic: topic/#, QoS: 0, Options: 0x00)
Client null received SUBACK
Subscribed (mid: 1): 0
Client null received PUBLISH (d0, q0, r1, m0, 'topic/pub', ... (22 bytes))
QoS 0 viestin toimitus
^C
C:\mosquitto>
```

Kuva 10. viestin julkaisu ja vastaanotto QoS 0 -palvelutasoa käyttäen.

4.2.2. Viestiliikenteen mittaus palvelutaso QoS 0-käytöstä

QoS 0-palvelutason viestien lukumäärää ja lähetettyjen/vastaanotettujen viestien pituuksia tutkittiin käyttämällä Wireshark ohjelmaa. Kuvan 11 yläosassa havaitaan pakettien määrät ja tarkat lähetysajat on merkitty sekunteina. Lisäksi mittauksessa on käytetty suodattimena mqtt:tä rajamaan näytölle vain MQTT-protokollan liikennettä. MQTT:n eri viestipakettien pituudet on nähtävissä kuvassa 11, esimerkiksi viestin julkaisussa (Publish Message) viestipaketin pituus on 99 tavua. Viestien pituudet ovat vakiot riippumatta palvelutasosta, pois lukien Publish Message viesti, jonka pituus riippuu lähetetyn sanoman pituudesta.

Kuvan alaosassa nähdään, että Mosquitto on lähettänyt 9 pakettia, joista 6:n paketin pituudet ovat väliltä 40-79 tavua ja edustavat siten prosenttiosuudeltaan 66,67 %, loput 3 pakettia ovat pituudet väliltään 80-159 tavua ja edustavat kokonaismäärästä 33,33 %. Kuvasta havaitaan myös, että lyhyiden pakettien pituuksien minimiarvo on 66 tavua ja pituuksien maksimiarvo on 78 tavua. Pitkien pakettien pituuksien minimi arvo on 84 tavua ja maksimi pituuksien arvo on 99 tavua.



Kuva 11. Pakettien määrät ja koot.

4.2.3. Viestin lähetys QoS 1 -palvelutasolta

Kuva 12 yläosassa havaitaan, että julkaisija on lähettänyt CONNECT, PUBLISH ja DISCONNECT -viestit, sekä vastaanottanut CONNACK-viestin lisäksi PUBACK-kuittausviestin sanoman perillemenon vakuudeksi, kuten kappaleessa 3.5.2 kuvassa 7 on esitetty. Viestin lähetyksessä on käytetty palvelutaso QoS 1:stä. Vastaanottaja ottaa vastaan viestin välittäjältä QoS 1 -palveluntasolta. Kuvan 12 alaosassa viestin vastaanottaja on kuitannut saamansa viestin lähettämällä välittäjälle PUBACK-viestin, kuten kappaleen 3.5.2:n kuvassa 7 on esitelty ja selostettu viestien merkitykset.

```
C:\mosquitto>mosquitto_pub -h localhost -t topic/pub -m "QoS 1 viestin toimitus" -r -i client -d -q 1
Client client sending CONNECT
Client client received CONNACK (0)
Client client sending PUBLISH (d0, q1, r1, m1, 'topic/pub', ... (22 bytes))
Client client received PUBACK (Mid: 1, RC:0)
Client client sending DISCONNECT

C:\mosquitto>
```

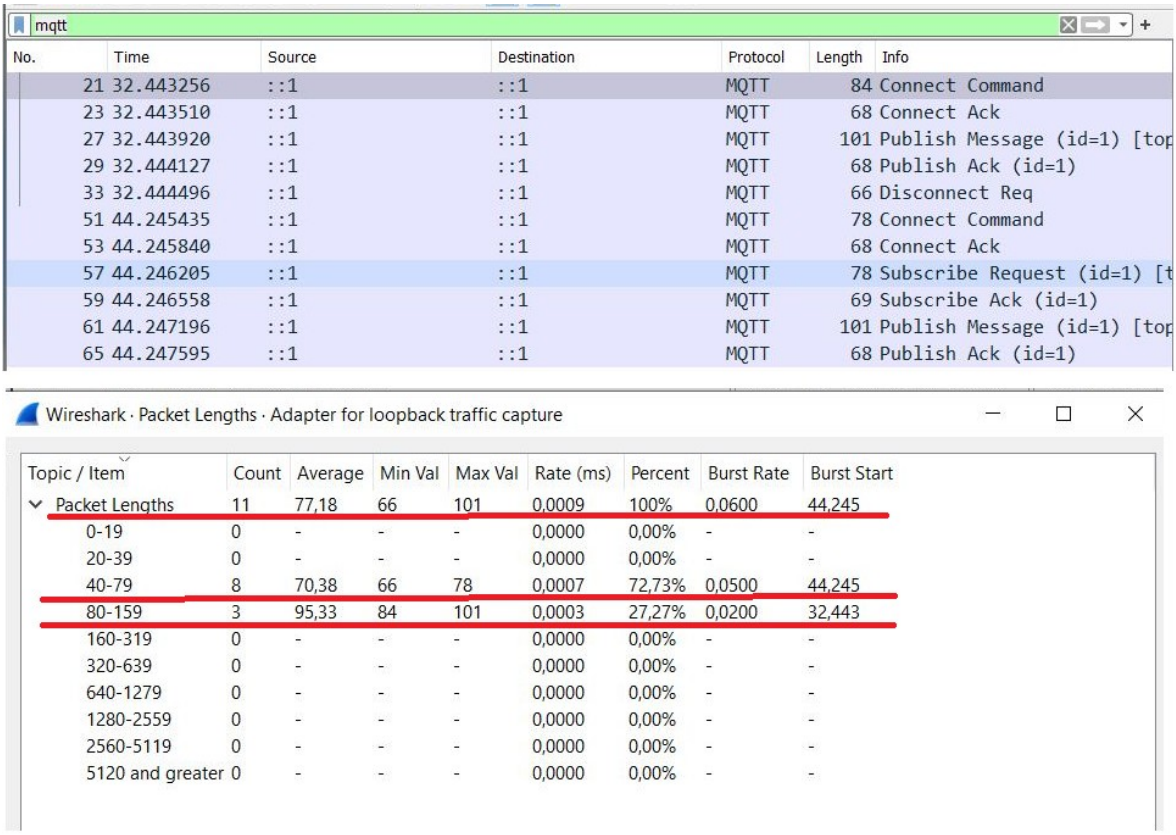
```
C:\mosquitto>mosquitto_sub -h localhost -t topic/# -d -q 1
Client null sending CONNECT
Client null received CONNACK (0)
Client null sending SUBSCRIBE (Mid: 1, Topic: topic/#, QoS: 1, Options: 0x00)
Client null received SUBACK
Subscribed (mid: 1): 1
Client null received PUBLISH (d0, q1, r1, m1, 'topic/pub', ... (22 bytes))
Client null sending PUBACK (m1, rc0)
QoS 1 viestin toimitus
^C
C:\mosquitto>
```

Kuva 12. Viestin lähetys ja vastaanotto palvelutaso QoS 1 käyttämällä.

4.2.4. Viestiliikenteen mittaus palvelutasolla QoS 1

Wiresharkilla suoritetussa verkkoliikenteen mittauksessa havaitaan, että mitattujen pakettien määrä on suurempi QoS 1 palvelutasolla, kuin aikaisemmassa QoS 0-palvelutason verkkoliikenteen mittauksessa. Pakettien määrän kasvu johtuu siitä, että QoS 1 -palvelutasolla lähetetään kaksi kappaletta kuittauksia, kuten kappaleessa 3.5.2 kuvassa 7 on kuvattu.

Kuvan 13 yläosassa kuittausviestit ovat nimeltään Publish Ack (id=1). Kuva 13 alaosassa nähdään, että Mosquitto on lähettänyt 11 pakettia, joista 8:n paketin pituudet ovat väliltä 40-79 tavua ja loput 3 pakettia ovat pituuksien väliltä 80-159 tavua. Kuvasta havaitaan myös, että lyhyiden pakettien minimiarvon pituus on 66 tavua ja maksimiarvon pituus on 78 tavua, eli arvot ovat samat kuin palvelutasolla QoS 0. Pitkien pakettien pituuksien minimi arvo on 84 tavua ja maksimi pituuksien arvo on 101 tavua. Alemmasta kuvasta voidaan havaita, että pakettien kokonaismäärästä 8 pakettia on 72,73 % ja loput 3 pakettia kokonaismäärästä 27,27 %. Lyhyiden pakettien prosenttiosuus kasvoi, koska edellä mainitut Publish Ack (id=1) ovat lyhyitä paketteja.



mqtt

No.	Time	Source	Destination	Protocol	Length	Info
21	32.443256	::1	::1	MQTT	84	Connect Command
23	32.443510	::1	::1	MQTT	68	Connect Ack
27	32.443920	::1	::1	MQTT	101	Publish Message (id=1) [top
29	32.444127	::1	::1	MQTT	68	Publish Ack (id=1)
33	32.444496	::1	::1	MQTT	66	Disconnect Req
51	44.245435	::1	::1	MQTT	78	Connect Command
53	44.245840	::1	::1	MQTT	68	Connect Ack
57	44.246205	::1	::1	MQTT	78	Subscribe Request (id=1) [t
59	44.246558	::1	::1	MQTT	69	Subscribe Ack (id=1)
61	44.247196	::1	::1	MQTT	101	Publish Message (id=1) [top
65	44.247595	::1	::1	MQTT	68	Publish Ack (id=1)

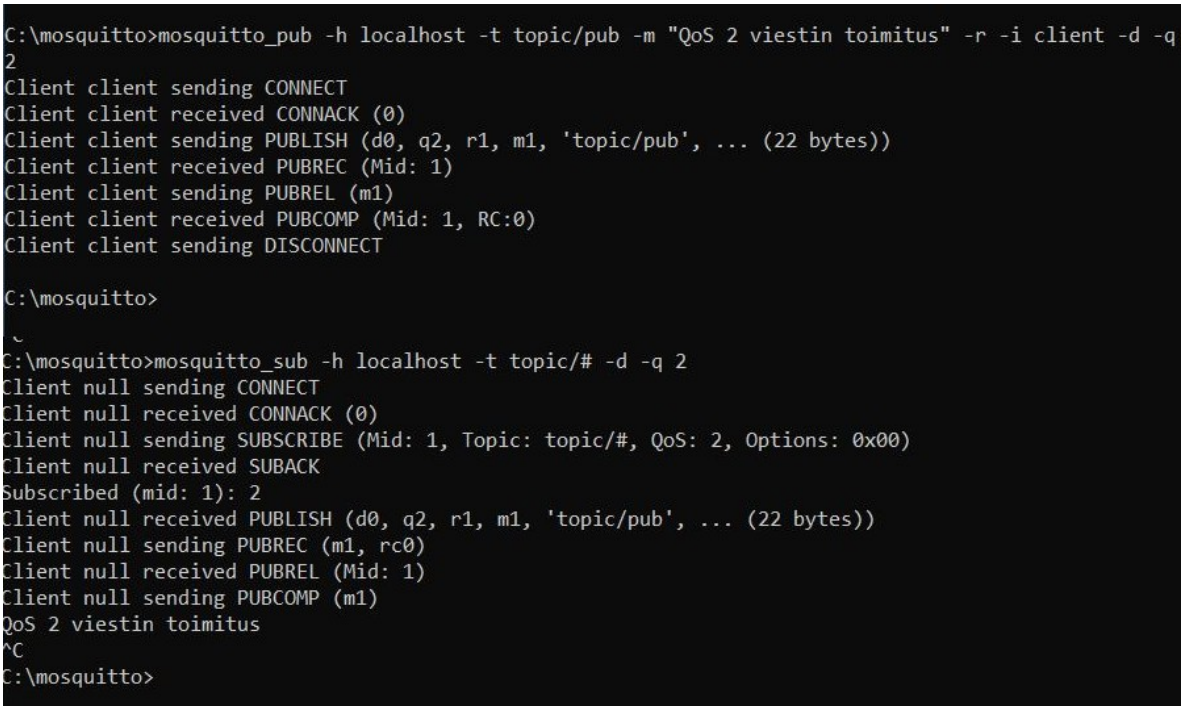
Wireshark · Packet Lengths · Adapter for loopback traffic capture

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
Packet Lengths	11	77,18	66	101	0,0009	100%	0,0600	44,245
0-19	0	-	-	-	0,0000	0,00%	-	-
20-39	0	-	-	-	0,0000	0,00%	-	-
40-79	8	70,38	66	78	0,0007	72,73%	0,0500	44,245
80-159	3	95,33	84	101	0,0003	27,27%	0,0200	32,443
160-319	0	-	-	-	0,0000	0,00%	-	-
320-639	0	-	-	-	0,0000	0,00%	-	-
640-1279	0	-	-	-	0,0000	0,00%	-	-
1280-2559	0	-	-	-	0,0000	0,00%	-	-
2560-5119	0	-	-	-	0,0000	0,00%	-	-
5120 and greater	0	-	-	-	0,0000	0,00%	-	-

Kuva 13. Verkkoliikenteestä mitatut paketit ja niiden jakautuminen eri pituuksiin.

4.2.5. Viestin lähetys QoS 2 -palvelutasolta

Viestin julkaisussa ja vastaanottamisessa on testattu vielä QoS 2 -palvelutason käyttämistä. Samassa mittauksessa mitattiin myös verkkoliikenteen käytön määrää. Mittauksessa havaittiin, että QoS 2 -palvelutaso käyttää enemmän verkkoliikenteen kaistaa, kuin aikaisemmat palvelutasot. Kuvan 14 yläosassa havaitaan, että viesti on julkaistu QoS 2 -palvelutasolla. Julkaisija vastaanottaa vastaussanoman PUBREC, jonka jälkeen julkaisija lähettää PUBREL sanoman ja vastaanottaa PUBCOMP-sanoman. Kuvan 14 alemmassa osassa nähdään vastaanottajan vastaanottavan viestin välittäjältä QoS 2 -palvelutasolta. Vastaanottaja lähettää PUBCOMP-sanoman välittäjälle onnistuneesta viestin vastaanotosta. Edellä mainittujen sanomien lähetyksistä on kuvattu kappaleessa 3.5.2 kuvassa 8.

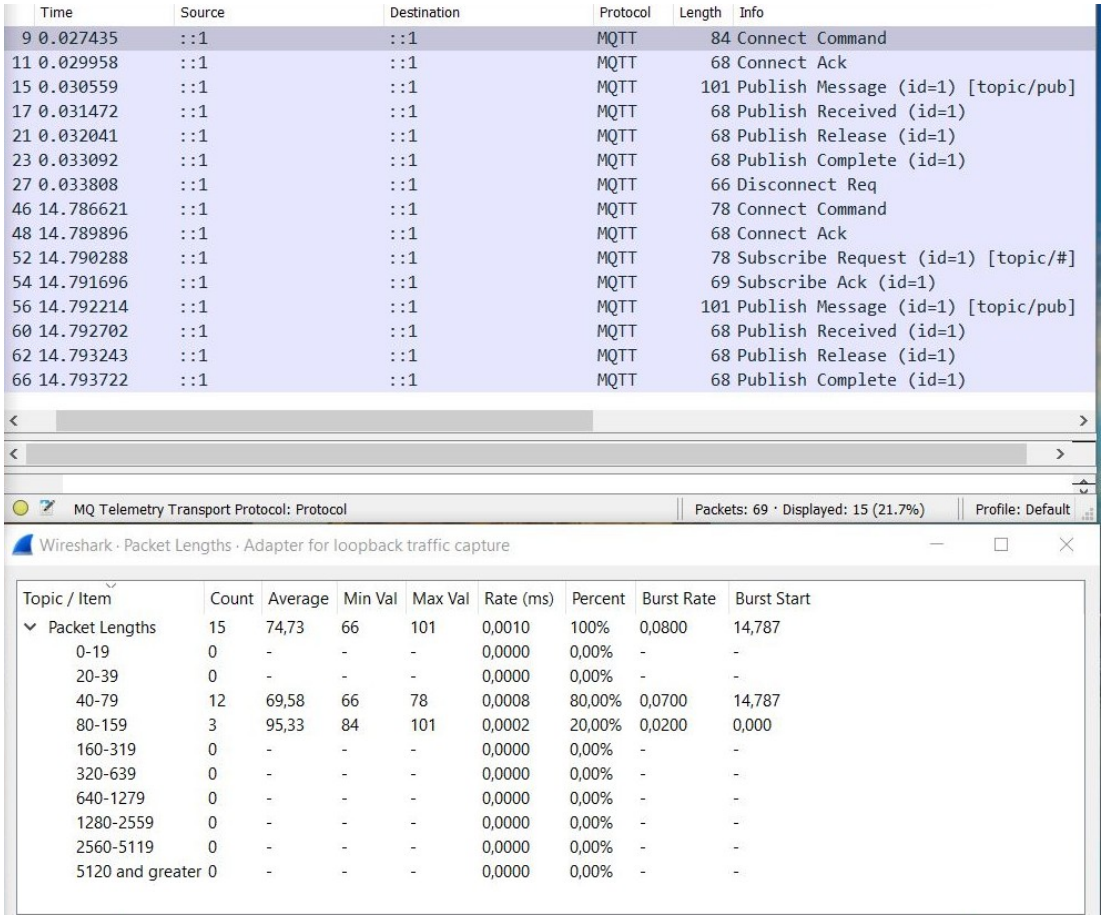


Kuva 14. Viestin julkaisu ja vastaanotto QoS 2 palvelutasolta.

4.2.6. Viestiliikenteen mittaus palvelutaso QoS 2 käytöstä

Verkkoliikenteen mittauksella on testattu viestin lähetyksen ja vastaanottamisen ominaisuuksia käytettäessä palvelutasoa QoS 2. Verkkoliikenteen mittauksessa havaitaan, että mitattujen pakettien määrä on suurempi QoS 2 -palvelutasolla viestien lähettämisessä ja vastaanottamisessa vertailtaessa aikaisemmissa QoS 0 ja QoS 1 -palvelutasojen pakettien määrään verkkoliikenteessä. Kappaleessa 3.5.2 A. Viswanathan on todennut samoin, että QoS 0-tasolla on vähiten lähetyksiä. Kuittausviestit ovat kuvan 15 yläosassa nimillä Publish Received (id=1) kaksi kappaletta, Publish Release (id=1) kaksi kappaletta, Publish Complete (id=1) kaksi kappaletta, sekä kaksi kappaletta Connect Ack, viimeksi mainitut kuittausviestit löytyvät myös palvelutaso QoS 1:n paketeista.

Kuvassa 15 alaosassa nähdään, että Mosquitolta on lähtenyt 15 pakettia, joista 12:n paketin pituudet ovat väliltä 40-79 tavua ja lopåujen kolmen pakettia pituudet ovat väliltä 80-159 tavua. Kuvasta havaitaan myös, että lyhyiden pakettien minimiarvon pituus 66 tavua ja maksimiarvon pituus on 78 tavua. Pitkien pakettien pituuksien minimi arvo on 84 tavua ja maksimi pituuksien arvo on 101 tavua. Alemmasta kuvasta voidaan havaita, että pakettien kokonaismäärästä 12 pakettia on 80 % ja loput kolme pakettia kokonaismäärästä 20 %. Lyhyiden pakettien osuus kasvaa prosenteissa enemmän, kuin QoS 0 ja Qos 1 -palvelutasoissa. Ilmiö johtuu siitä, että palvelutaso QoS 2:lla lähetetään enemmän kuittausviestejä. Sama asia on todettu myös kappaleessa 3.5.2 kuvassa 8.



Time	Source	Destination	Protocol	Length	Info
9 0.027435	::1	::1	MQTT	84	Connect Command
11 0.029958	::1	::1	MQTT	68	Connect Ack
15 0.030559	::1	::1	MQTT	101	Publish Message (id=1) [topic/pub]
17 0.031472	::1	::1	MQTT	68	Publish Received (id=1)
21 0.032041	::1	::1	MQTT	68	Publish Release (id=1)
23 0.033092	::1	::1	MQTT	68	Publish Complete (id=1)
27 0.033808	::1	::1	MQTT	66	Disconnect Req
46 14.786621	::1	::1	MQTT	78	Connect Command
48 14.789896	::1	::1	MQTT	68	Connect Ack
52 14.790288	::1	::1	MQTT	78	Subscribe Request (id=1) [topic/#]
54 14.791696	::1	::1	MQTT	69	Subscribe Ack (id=1)
56 14.792214	::1	::1	MQTT	101	Publish Message (id=1) [topic/pub]
60 14.792702	::1	::1	MQTT	68	Publish Received (id=1)
62 14.793243	::1	::1	MQTT	68	Publish Release (id=1)
66 14.793722	::1	::1	MQTT	68	Publish Complete (id=1)

MQ Telemetry Transport Protocol: Protocol | Packets: 69 · Displayed: 15 (21.7%) | Profile: Default

Wireshark · Packet Lengths · Adapter for loopback traffic capture

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
Packet Lengths	15	74,73	66	101	0,0010	100%	0,0800	14,787
0-19	0	-	-	-	0,0000	0,00%	-	-
20-39	0	-	-	-	0,0000	0,00%	-	-
40-79	12	69,58	66	78	0,0008	80,00%	0,0700	14,787
80-159	3	95,33	84	101	0,0002	20,00%	0,0200	0,000
160-319	0	-	-	-	0,0000	0,00%	-	-
320-639	0	-	-	-	0,0000	0,00%	-	-
640-1279	0	-	-	-	0,0000	0,00%	-	-
1280-2559	0	-	-	-	0,0000	0,00%	-	-
2560-5119	0	-	-	-	0,0000	0,00%	-	-
5120 and greater	0	-	-	-	0,0000	0,00%	-	-

Kuva 15. Verkkoliikenteessä mitattujen pakettien määrä Mosquitton käyttäessä QoS 2 -palvelutasoa viestien lähettämiseen ja vastaanottamiseen.

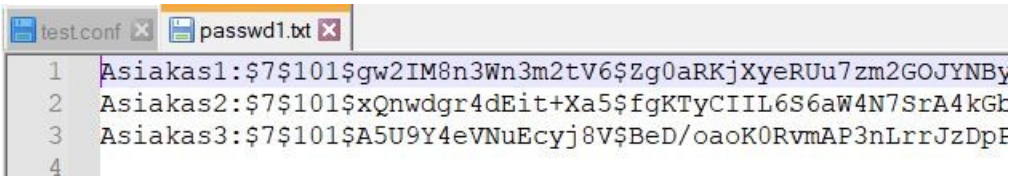
Taulukko 6:ssa on vertailtu eri palvelutasojen tuottamien pakettien määrä, pakettien pituuksien keskiarvo, datan lähetysnopeudet ja purkausnopeudet, jolla viitataan tietyllä aikavälillä laskettujen pakettien määrään, joita verrataan aikaväleihin aikaikkunassa. Oletuksena purskeet tunnistetaan viiden millisekunnin välein ja sitä verrataan aikaikkunaan. Pakettien pituuksien keskiarvo pienenee palvelutason kasvaessa. Tämä johtuu siitä, että korkeammalla palvelutasolla lähetetään enemmän lyhyitä kuittausviestejä kuin alemmalla palvelutasolla.

Palvelutaso	Paketti/kpl	Pituuden keskiarvo	Nopeus (ms)	Purkautuva nopeus
QoS 0	9	78.78	0.0002	0.0500
QoS 1	11	77.18	0.0009	0.0600
QoS 2	15	74.73	0.0010	0.0800

Taulukko 6. Verrataan eri palvelutasojen ominaisuuksia.

4.3. Tietoturvan varmentaminen käyttäjätunnuksella ja salasanalla

Käyttäjätunnuksen ja salasanan käyttö vaatii muutoksia konfiguraatiotiedostossa. Muutoksissa välittäjälle luodaan tiedosto käyttäjätunnuksista ja niiden salasanoista. Kappaleessa 3.5.1 mainitaan A. Viswanathanin [16 s. 23] todenneen saman käyttäjätunnuksen ja salasanan käytöstä. Lisätietoturvaa saadaan kryptaamalla salasanat Mosquittolla. Kuvassa 16 on kryptauksessa käytetty Hash-funktio menetelmää. Salasanan kryptauksen jälkeen salasana ei ole selkokielistä, joten luvaton kirjautuminen vaikeutuu. Testissä havaittiin, että tiedostosta kopioidulla, kryptatulla salasanalla kirjautuminen ei onnistunut. Kuvassa 16 havaitaan, että kolmen käyttäjätunnuksen salasanat ovat kryptattuja, kryptaus näkyy satunnaisina merkkeinä.

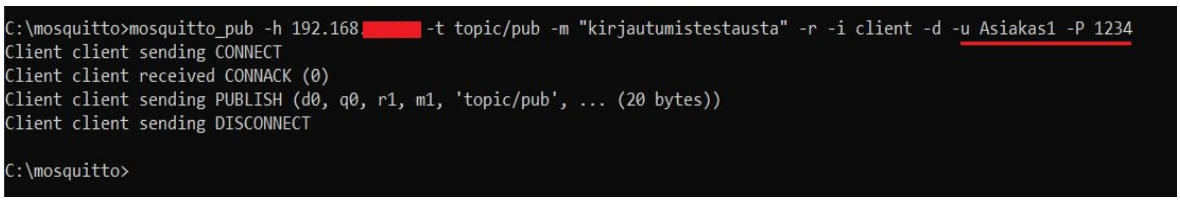


Kuva 16. Luotujen asiakkaiden käyttäjätunnukset ja niiden salasanat ovat kryptattu.

Yksi menetelmä tietoturvan lisäämiseksi on käyttää varmennettua yhteyttä. Kuvassa 17 viesti lähetetään välittäjälle, asiakas saa vastauksen CONNACK (5), josta havaitaan yhteysvirhe “Connection error”. Virheilmoituksessa lukee “not authorised”, joka tarkoittaa, että vastaanottajaa ei ole tunnistettu ja ERROR ilmoitus kertoo, että yhteys on epäonnistunut. Alussa kerrottiin, että tässä testauksessa halutaan suorittaa kahden asiakkaan välistä viestintää.

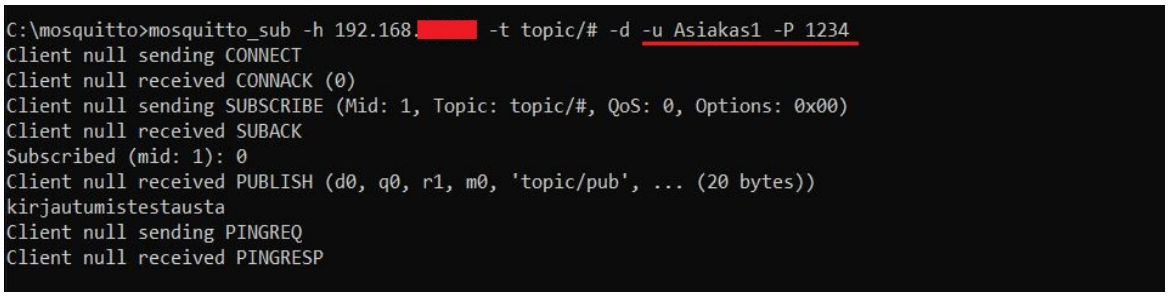
Kuva 17. Epäonnistunut viestin lähetys.

Halutun vastaanottajan varmentamiseksi lisätään komentoikkunassa viestin aiheen jälkeen halutun asiakkaan käyttäjätunnus ja salasana. Kuvassa 18 havaitaan, että komennon perään on lisätty vastaanottajan käyttäjätunnus Asiakas2 ja salasana. Tämän jälkeen viestin lähettäminen on onnistunut. Tämä on tietoturvallisuuden kannalta yksi menetelmistä, jossa viestin lähettämistä varten tarvitaan varmennettu vastaanottaja. Kryptatun salasanan ja TLS:n ominaisuuksia on käyty läpi kappaleessa 4.4.2.



Kuva 18. Viesti lähetetty varmennettuna.

Viestin avaamiseksi tarvitaan oikea käyttäjätunnus ja salasana. Asiakas tilaa välittäjältä aiheen perusteella viestejä. Kuvassa 19 vastaanottaja avaa viestin varmennuksen avulla, joka on tietoturvallisuuden kannalta varsin tärkeä, jotta viestin saa oikea asiakas ja eikä muut pääse lukemaan omilla tunnuksillaan.

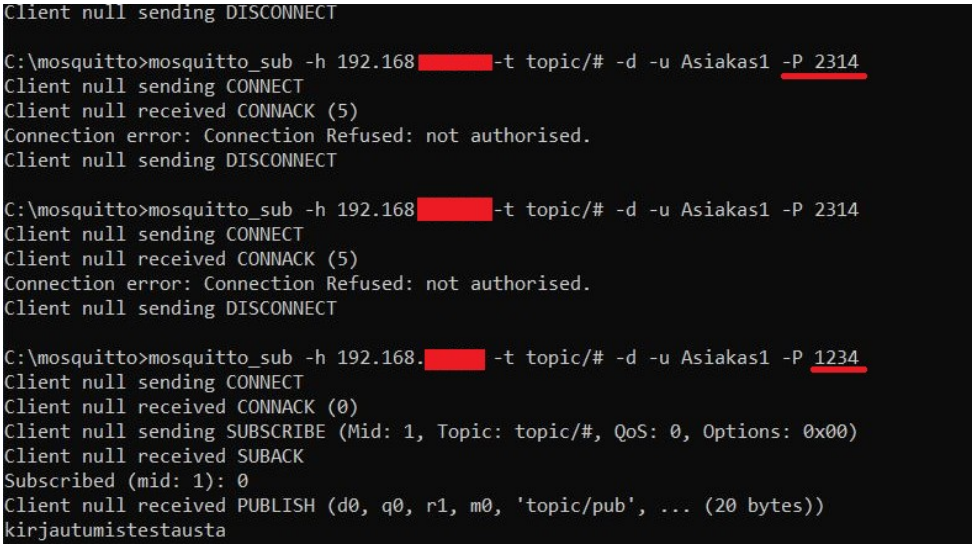


Kuva 19. Haluttu vastaanottaja saa viestin.

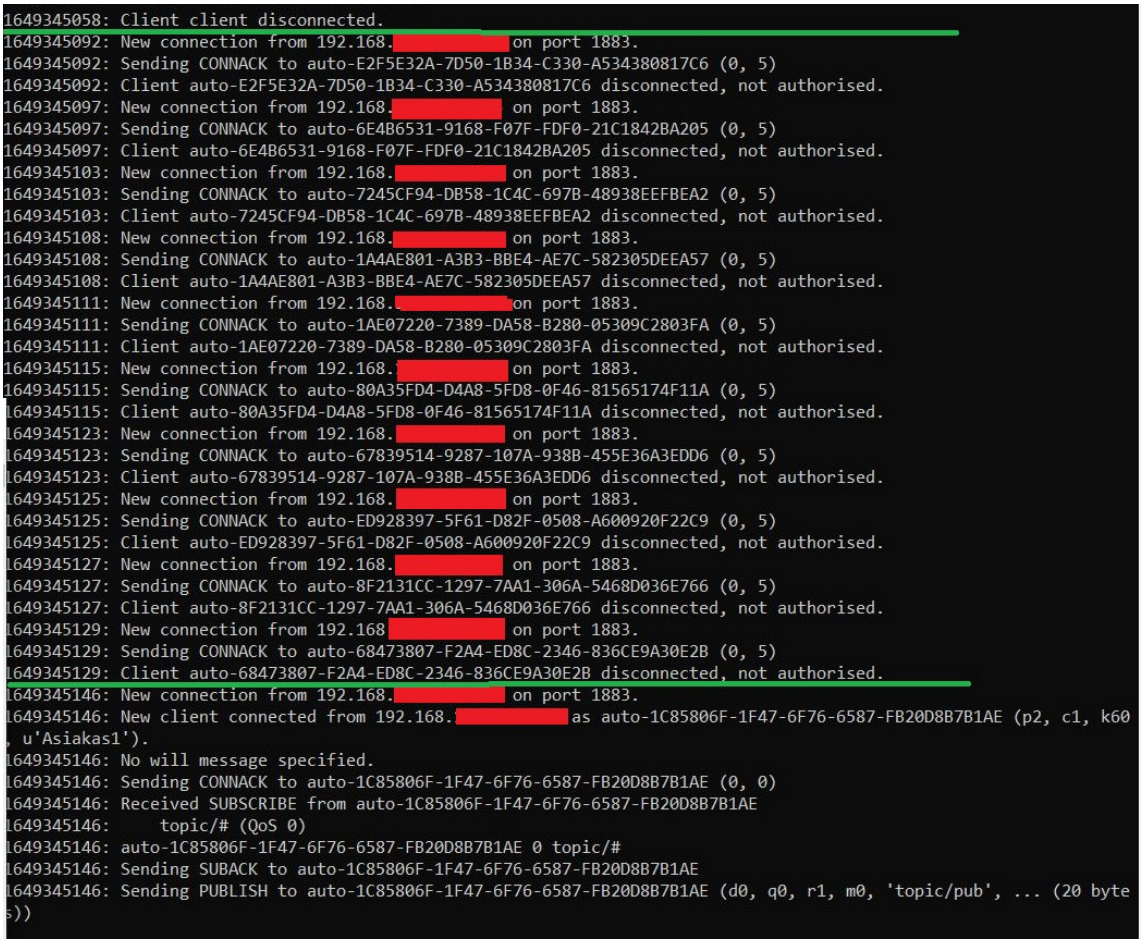
4.4. Haavoittuvuuksien todentaminen MQTT:ssä

Kappaleessa 3.5.1 kerrotaan G. Nebbionen & M. C. Calzarossan [13 s. 5] raportoineen MQTT- protokollan haavoittuvuudesta. Julkaisuissa mainittiin, että MQTT- protokokolla ei rajoita todennusyritysten kertoja. Vahvistaakseen julkaisu tekseissä mainittuja ominaisuuksia testattiin todennusyrityskertoja, kirjoittamalla 10 kertaa halutun asiakkaan salasanan väärin, eikä ohjelma yritä edes estää todennuskertojen yrityksiä.

Kuvassa 20 havaitaan todennusyritysten kertojen määrä. Kuvassa näkyy väärällä salasanalla 10 todennuskertaa. Yhdennellätoista todennuskerralla kirjoitettiin asiakkaan oikea salasana. Yhteyden luonti onnistui ja asiakas sai halutun viestin avattua oikealla salasanalla. Kuvassa 21 lokitietoikkunassa havaitaan 11 todennusyrityskertaa, joista 10 ilmoittaa “ei valtuutuksia”.



Kuva 20. Todennusyrityskertoja suoritettu väärällä ja oikealla salasanalla



Kuva 21. Lokitietoikkunassa havaitaan kirjautumisyrityskertojen määrän olevan 11.

Salasanan selvittämisen onnistuminen riippuu paljon salasanasta, sen pituudesta ja salasanassa käytetyistä merkeistä. Alla olevassa taulukossa on kuvattu murtautumisaikaa salasanamurto-ohjelman avulla.

Merkkimäärä	numeroina	Pienet kirjaimet	Isot ja pienet kirjaimet	Numerot, pienet ja isot kirjaimet	Numerot, pienet ja isot kirjaimet, erikoismerkit
5	välittömästi	välittömästi	välittömästi	välittömästi	välittömästi
7	välittömästi	välittömästi	25 s	1 min	6 min
9	välittömästi	2 min	19 h	3 päivää	3 viikkoa
10	välittömästi	58 min	1 kk	7 kk	1 v
12	25 s	3 viikkoa	300 vuotta	2 000 vuotta	34 000 v
14	41 min	51 v	8 000 v	9 000 000 v	2 0000 0000 v

Taulukko 7. Salasanan merkkien määrän ja lajin vaikutus salasanan murtamisaikaan.[12]

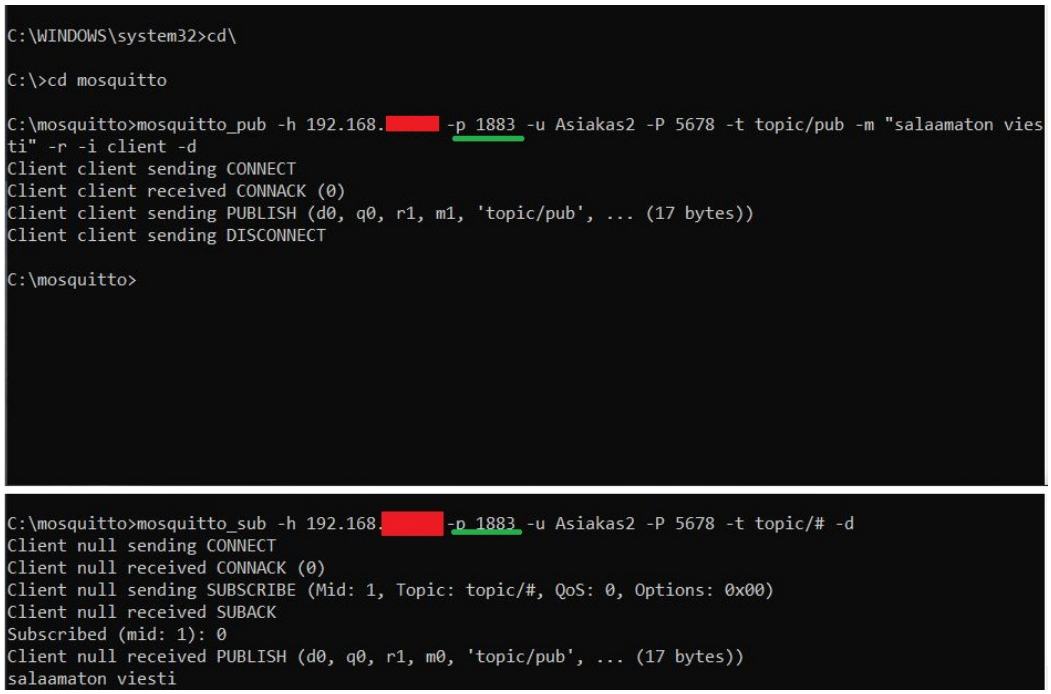
4.5. Viestiliikenteen suojaaminen TLS:n avulla

Tässä testissä testattiin TLS:n toimivuutta, joka on tietoturvan kannalta varsin tärkeä. Salattu viestiliikenne ohjataan portin 8883 kautta, koska kyseinen portti tukee TLS-standardia, kuten kappaleessa 3.1 A. Viswanathan on todennut saman asian [16 s. 22-23]. Käytettäessä TLS-salausta, ulkopuoliset eivät pysty avaamaan viestien sisältöjä. Kuvassa 22 esitetään Mosquitton lokitietoikkuna, josta voidaan havaita, että käytössä on suojaamaton portti 1883. Kuvan 23 alaosasta havaitaan, että asiakas on saanut viestin portin 1883 kautta, joka on suojaamaton ja ulkopuolinen voisi nähdä viestin sisällön.

```
1649593709: New connection from 192.168.[redacted] on port 1883.
1649593709: New client connected from 192.168.[redacted] as client (p2, c1, k60, u'Asiakas2').
1649593709: No will message specified.
1649593709: Sending CONNACK to client (0, 0)
1649593709: Received PUBLISH from client (d0, q0, r1, m0, 'topic/pub', ... (17 bytes))
1649593709: Received DISCONNECT from client
1649593709: Client client disconnected.
1649593859: New connection from 192.168.[redacted] on port 1883.
1649593859: New client connected from 192.168.[redacted] as auto-EC9F32FB-2466-DBF0-0666-3B46831371F8 (
p2, c1, k60, u'Asiakas2').
1649593859: No will message specified.
1649593859: Sending CONNACK to auto-EC9F32FB-2466-DBF0-0666-3B46831371F8 (0, 0)
1649593859: Received SUBSCRIBE from auto-EC9F32FB-2466-DBF0-0666-3B46831371F8
1649593859:     topic/# (QoS 0)
1649593859: auto-EC9F32FB-2466-DBF0-0666-3B46831371F8 0 topic/#
1649593859: Sending SUBACK to auto-EC9F32FB-2466-DBF0-0666-3B46831371F8
1649593859: Sending PUBLISH to auto-EC9F32FB-2466-DBF0-0666-3B46831371F8 (d0, q0, r1, m0, 'topic/pub', .
.. (17 bytes))
```

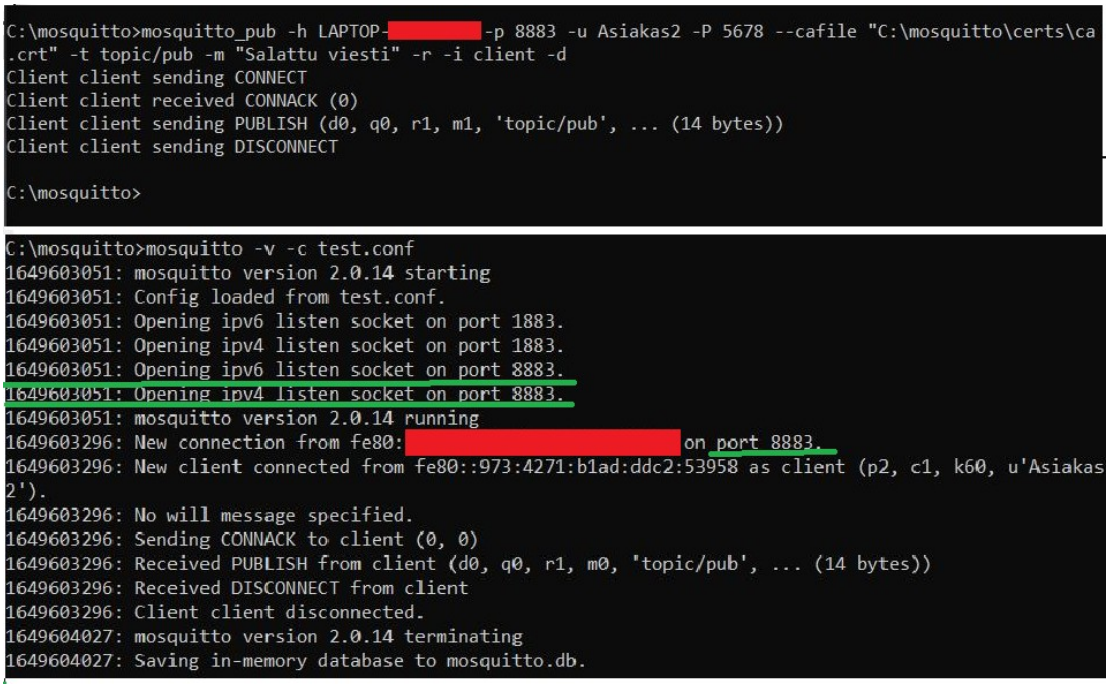
Kuva 22. Mosquitton lokitietoikkuna näkymä portin 1883 käytöstä.

12 Alkuperäinen taulukko osoitteessa https://itigic.com/fi/how-long-does-it-take-to-hack-or-crack-a-password/.



Kuva 23. Lähetetään ja vastaanotetaan viesti käyttämällä suojaamatonta porttia 1883.

Toisessa testissä lähetettiin viesti, jossa oli TLS-salaus käytössä. Kuunteluporttina oli tällä kertaa käytössä TLS-salausta tukeva portti 8883, jolloin ulkopuoliset eivät voi lukea tietoliikenteessä kulkevaa dataa. Kuva 24 yläosassa havaitaan asiakkaan lähettäneen viestin, joka käyttää porttia 8883 ja alaosassa olevassa lokitietoikkunassa nähdään yhteydenotossa käytetyn porttia 8883.

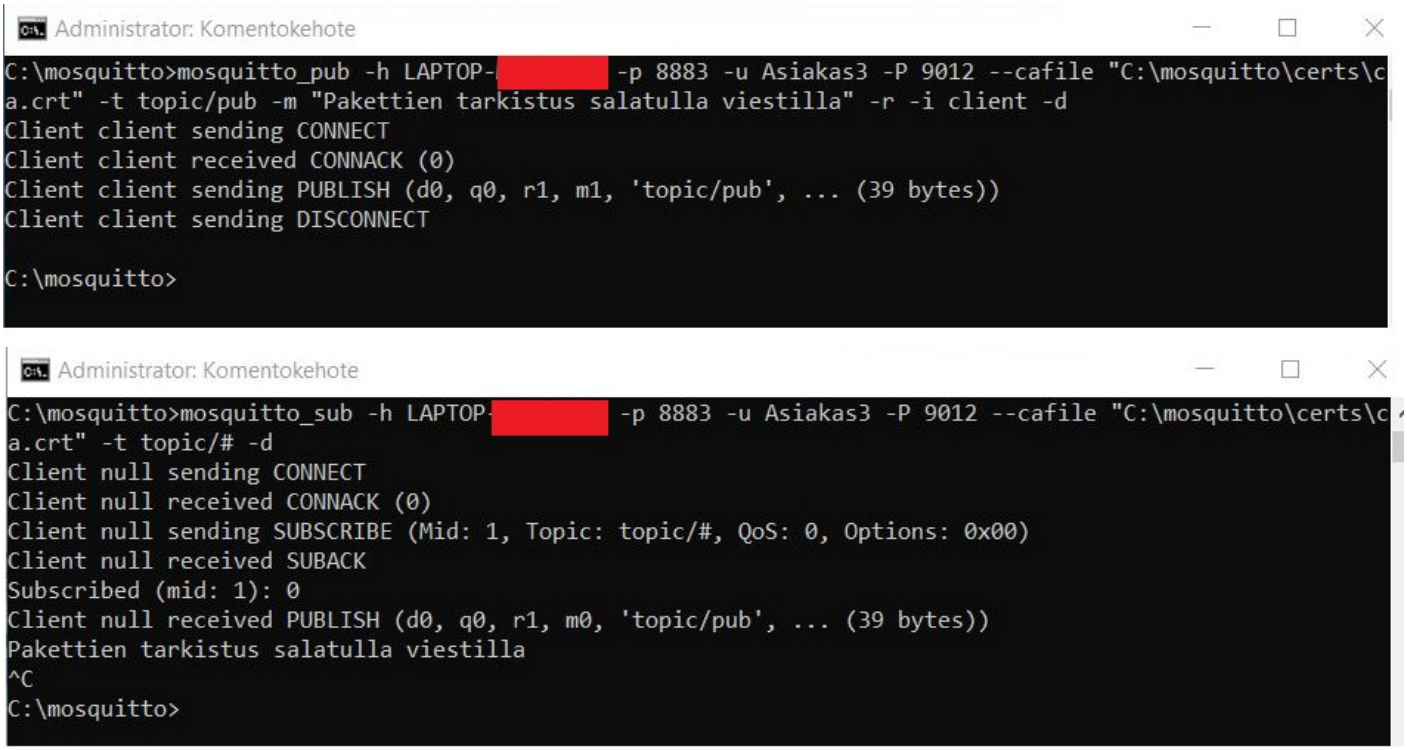


Kuva 24. Viestin lähettämiseen on käytetty viestiliikenteessä TLS-salausta tukevaa porttia 8883.

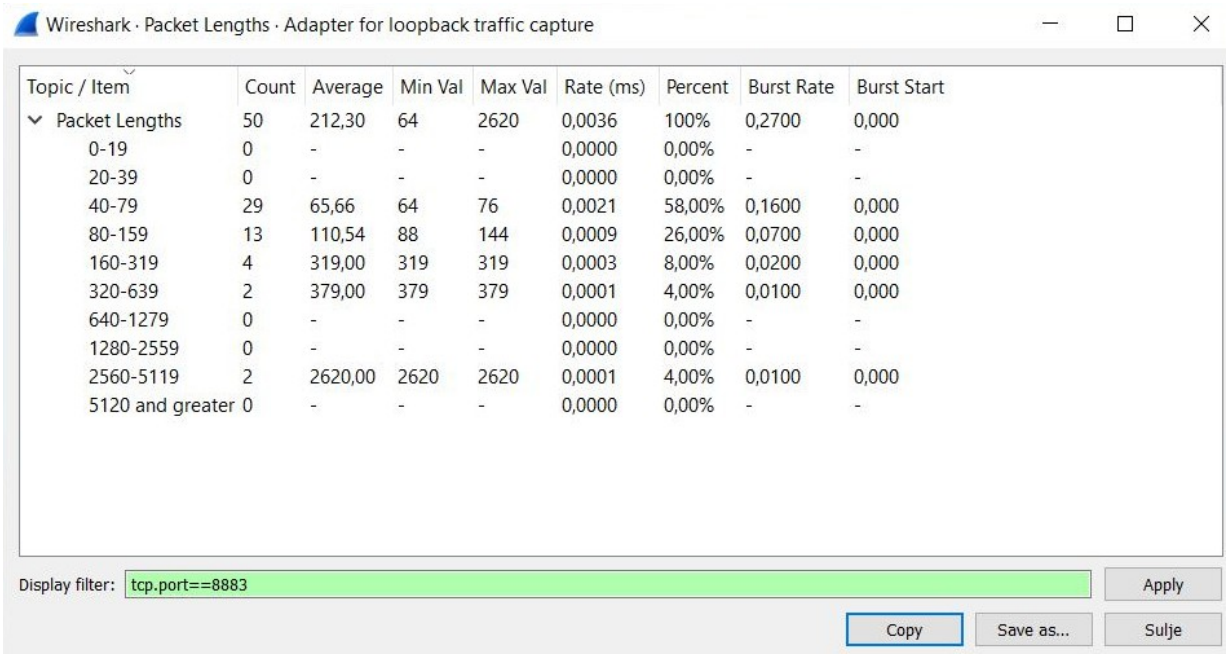
4.5.1. TLS:n käyttämän liikenteen mittaaminen

Edellisen lisäksi suoritettiin lisätestejä verkkoliikenteen mittauksessa, kun TLS oli käytössä ja samalla tehtiin havaintoja siitä, miten TLS:n käyttö eroaa tavallisen suojaamattoman viestin lähettämisestä. Aikaisemmissa verkkoliikenteen mittauksessa viestien lähettämisessä käytettiin suojaamatonta porttia 1883 ja tässä uudessa mittaustestissä käytettiin porttia 8883 ja suodatettiin monitorille ilmestyviä tuloksia. Suodattaminen toteutettiin lisäämällä suodattimeksi tcp.port = = 8883, jolla rajattiin näytölle liikenteestä tulostuvat viestipaketit, jotka käyttivät porttia 8883.

Kuvan 25 yläosassa havaitaan, että viestin lähettämiseen on käytetty edellä mainittua suojattua porttia ja alaosassa viesti on vastaanotettu. Merkittävämpi havainto havaitaan kuvassa 26, jossa on lista paketeista. Viestin lähettämisessä käytettiin QoS 0-palvelutasoa. Pakettien määrä oli tässä vaiheessa kasvanut merkittävästi verrattuna aikaisempiin viestiliikenteen mittauksissa saatuihin tuloksiin. Pakettien kasvumäärä havaitaan vertaamalla tulosta esimerkiksi kuvan 15 tuloksiin.

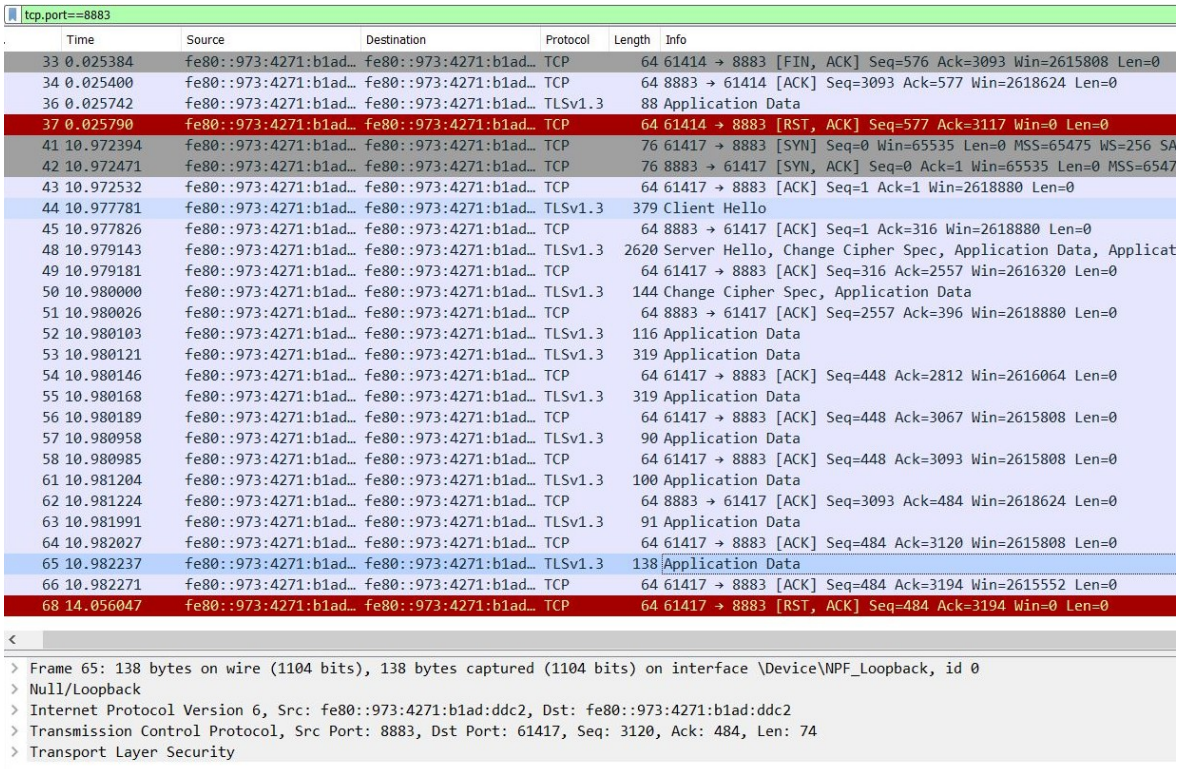


Kuva 25. Viesti lähetetty ja vastaanotettu suojattua porttia käyttämällä.



Kuva 26. Wiresharkilla mitattuja paketteja käyttämällä tcp.port = = 8883 suodatinta.

Tietoturvallisuuden osalta havaittiin, että TLS-suojauksen ansiosta Wiresharkilla ei voida analysoida viestiä. Kuvassa 27 käytetään suodattimena tcp.port = = 8883, jolla on eritelty paketit, jotka käyttävät porttia 8883.



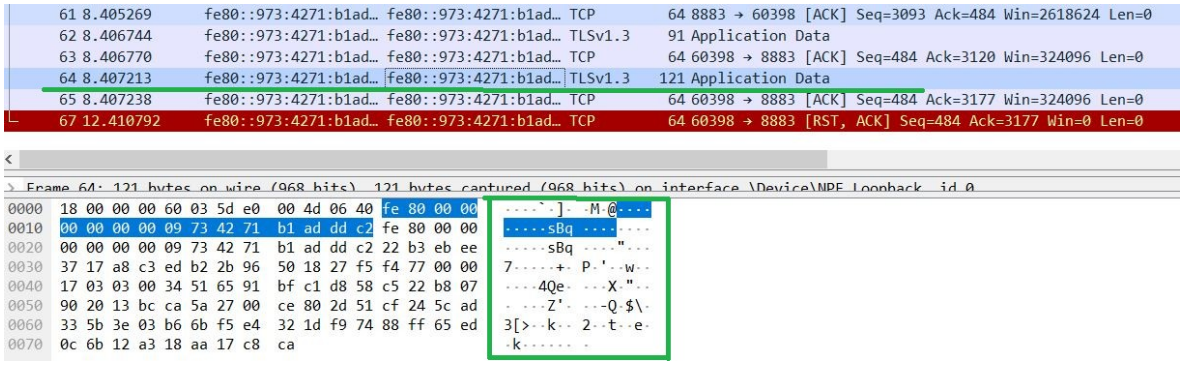
Kuva 27. Suodattimella suodatettu näkymään vain portin 8883 liikenne.

4.5.2. Viestien näkyvyys Wiresharkissa

Viestiliikenteessä on otettava huomioon muun muassa se, että suojaamattoman portin 1883 viestit eivät ole suojattu verkkoliikenteessä, jolloin niitä on mahdollista tarkastella esimerkiksi Wiresharkilla. Kappaleen 3.5.1:ssä G. Nebbione & M. C. Calzarossa mainitsivat julkaisussaan salaamattomasta viestinnästä. Kappaleessa mainitaan muun muassa, että viestit lähetetään selkokielisenä tekstinä. Kuitenkin viestit voidaan salata ja lähettää TLS:n standardoidun portin 8883 kautta, jolloin viestin sisältö tai käyttäjätunnus ja salasana eivät paljastu verkkoliikenteessä.

Kuva 28 alaosassa havaitaan, että viestin lähettämisessä on ohjattu viestit kulkemaan salaamattoman portin 1883 kautta, joten asiakkaan käyttäjätunnus ja salasana ovat nähtävissä Wiresharkissa, samoin viestin sisältö. Kuvassa 29 taas havaitaan, että samasta kohdasta otetussa kuvassa ei näy enää asiakkaan tunnusta ja salasanaa, eikä myöskään viestiä, koska salattu viesti on ohjattu kulkemaan suojatun portin 8883 kautta.

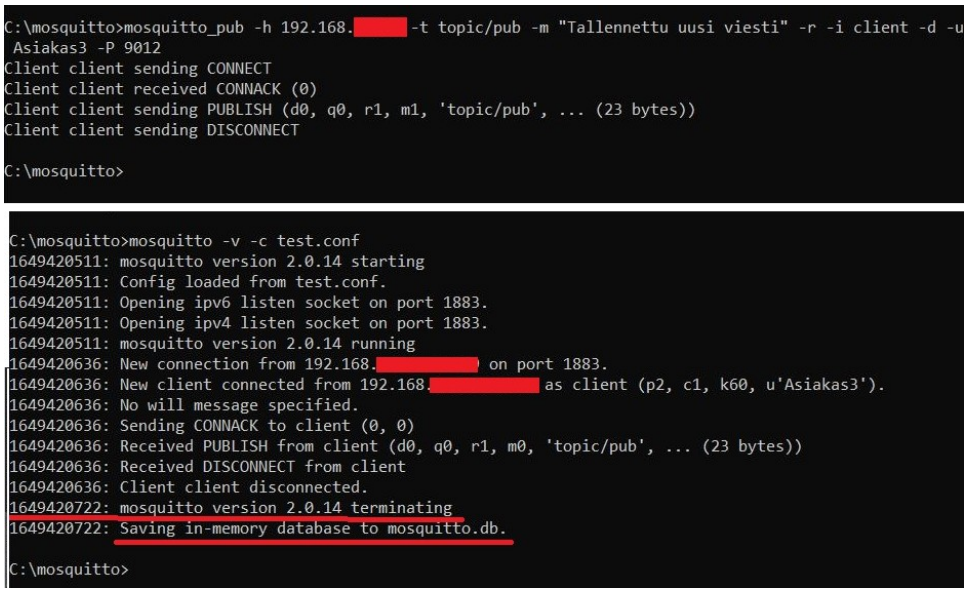
Kuva 28. Wiresharkissa näkee viestistä asiakkaan tunnuksen ja salasanan, sekä lähetetyn viestin sisällön, koska viestin lähetyksessä on käytetty salaamatonta porttia 1883.



Kuva 29. Wiresharkissa ei näy asiakkaan tunnusta ja salasanaa, eikä viestin sisältöä, koska käytössä on suojattu portti 8883.

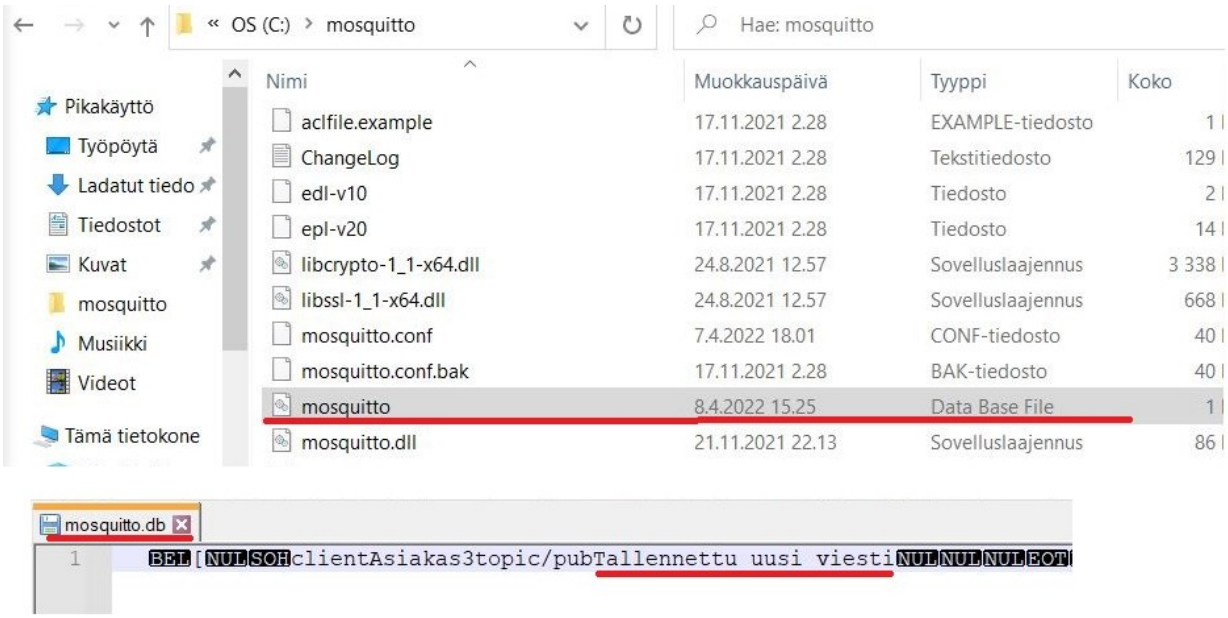
4.6. Viestien pysyvän tallennuksen merkitys

Seuraavassa testauksessa kokeiltiin MQTT-protokollan pysyvän tallennuksen ominaisuutta. Tätä MQTT:n ominaisuutta käytetään viestien pitämiseen tallessa, vaikka vastaanottajaan ei saada sillä hetkellä yhteyttä tai Mosquitto on pysähtynyt/sammunut ennen vastaanottajan viestipyyntöä. Tallennusominaisuus saadaan käyttöön lisäämällä se Mosquitton konfiguraatiotiedostoon. Kuvan 30 ylemmässä kuvassa havaitaan, että asiakas on lähettänyt viestin eteenpäin ja kuvan alaosassa on lokitietoikkuna, jossa Mosquitto on pysäytetty. Mosquitton lokitietoikkunasta havaitaan myös viesti (Saving in-memory database), joka ilmoittaa viestin tallentumista mosquitto.db tietokantatiedostoon. Tällä halutaan osoittaa viestin pysyvän tallennuksen merkitystä, eli viestiä ei ole menetetty. Vastaanottaja saa viestin seuraavalla kerralla, kun Mosquitto käynnistetään ja vastaanottaja lähettää viestipyynnön välittäjälle antaen käyttäjätunnuksen ja salasanan.



Kuva 30. Viesti on lähetetty ja tallennettu Mosquitton pysäytyksen jälkeen.

Kuvan 31 yläosassa havaitaan viestin tallentuneen mosquitto-kansioon mosquitto.db nimellä. Kuvan alaosassa nähdään tallennetun viestin sisältö, jonka vastaanottaja saa myöhemmin otettua vastaan. Kuvasta 31 havaitaan, että tietokantatiedostoon on tallennettu kryptaamaton viesti. Mikäli joku pääsee murtautumaan tietokoneelle, jossa broker sijaitsee, murtautuja voi päästä käsiksi .db tiedostoon ja voi lukea/muuttaa viestin sisällön esimerkiksi Notepad ++ tekstieditorilla. Kappaleessa 3.5.1 on kerrottu suojaamattomista palvelimista.



Kuva 31. Tietokantatiedosto tallennettu ja alaosassa tallennettu viesti näkyy.

5. TESTIEN TULOKSET

Ensimmäisessä testauksessa testattiin viestien lähettämistä ja vastaanottamista eri palvelutasoilla ja samalla mitattiin verkkoliikenteen käyttöä. Ensimmäinen testitulos QoS 0-palvelutasoa käyttäessä osoitti, että kyseessä oli yksinkertainen viestin lähettäminen välittäjälle ja viestin vastaanottaminen välittäjältä. Palvelutaso QoS 0 ei säilytä viestejä vaan hävittää ne lähetyksen jälkeen, kuten kappaleessa 3.5.2 kuva 5 esittää. Viestin julkaisussa/lähettämisessä yleiset MQTT-protokollan ominaisuudet toimivat, kuten MQTT:n ominaisuuksissa luvattiin.

Ensimmäisen testin aikana mitattiin myös viestidatan kulkua mittaamalla asiakkaan ja välittäjän välistä verkkoliikennettä käyttämällä Wireshark-ohjelmaa ja samalla selvitettiin, että kuinka paljon liikennettä QoS 0 palvelutaso käyttää. Mittaustuloksista selvisi, että datapaketteja QoS 0 lähettää 9 kappaletta. Niissä pakettien pituudet jakautuvat siten, että lyhyempien pakettien määrä on prosentuaalisesti enemmistö verratessa pidempien pakettien määrään. Sama koskee myös muita palvelutasoja.

Viestin lähettämisessä palvelutaso QoS 1:n ero QoS 0 oli havaittavissa PUBACK-kuittaussanoman lisäyksenä. PUBACK-sanoma on käytössä, kun viestin perille menolle oli asetettu kuittausmääre. Wireshark-mittauksessa pakettien pituuksien prosentuaalinen ero kasvoi QoS 1 palvelutasolla, verrattuna QoS 0 palvelutasoon. Tällöin lyhyempiä paketteja on QoS 1 tasolla on enemmän kuin QoS 0 tasolla. Tässä vaiheessa voidaan jo todeta, että mitä korkeammalla palvelutasolla viestit lähetetään, sen enemmän viestiliikenteessä lyhyiden pakettien pituuksien prosentti osuus kasvaa ja pidempien viestipakettien prosentuaalinen osuus laski. Kolmannessa testissä viestin lähetykseen ja vastaanottoon on käytetty QoS 2 -palvelutasoa, jota käytetään tärkeimpien viestien lähettämiseen.

Edelliseen palvelutasoon eroavaisuutena on, että QoS 2 palvelutasolla lähetetään PUBREL varmistussanoma ja vastaanotossa PUBCOMP sanoma, joka toimii kuittauksena viestin onnistuneesta vastaanottamisesta. Viestiliikenteen mittaamisessa havaittiin, että QoS 2-palvelutasolla datapakettien lyhyiden pakettien määrä on jo 80 % pakettien kokonaismäärästä. Ensimmäisten testituloksien loppuosassa laadittiin taulukko, jossa oli vertailtu kolmen palvelutasolta lähteneiden datapakettien määrää, pituuksien keskiarvoa, datan kulkunopeutta, sekä purkautumisnopeutta.

Toisessa testauksessa testattiin asiakkaan käyttäjätunnusten ja salasanojen toimivuutta ja todennettiin niiden toimivan, kuten MQTT:n julkaisijan sivustolla on luvattu. Vastaanottaakseen viestin toiselta asiakkaalta on tiedettävä oikea käyttäjätunnus ja salasana, jolloin vastaanottaja pääsee lukemaan viestiä. Käytettäessä suojaustoimenpiteenä pelkästään käyttäjätunnusta ja salasanaa, voidaan turvallisuutta parantaa käyttämällä erillistä verkkoa, jolla ei ole suoraa yhteyttä ulkoiseen verkkoon. Aiheesta on maininta kappaleessa 2.3.

Kolmas MQTT-testaus on jatko-osa edelliselle testille. Tässä testissä todettiin haavoittuvuus, joka löytyy MQTT:stä. Asiasta on maininta kappaleessa 3.5.1. Kyseessä on rajoittamattomat kirjautumisyrityskerrat, jonka mukaan voidaan arvata salasana niin monta kertaa, kunnes syöttää oikean salasanan. Kirjautumiskertoja testattiin 10 kertaa väärällä salasanalla. Tuloksena oli, että ohjelma ei yrittänyt estää kirjautumista. 11 kerralla kirjoitettiin salasana oikein, jolloin kirjautuminen onnistui normaalisti. Todennusyrityksiä ovat tutkineet G. Nebbione & M. C. Calzarossa.

Neljännessä testauksessa testattiin TLS-suojausta, jolla suojattiin verkkoliikennettä. Tulokset osoittavat suojauksen toimivan oikein, kun viestin lähettämisessä käytettiin suojattua porttia 8883, eikä suojaamatonta porttia 1883. Suojatun viestinnän avulla saadaan viestit kryptattua, eikä niitä ulkopuolinen pysty tulkitsemaan. Tässä neljännessä testissä tehtiin kaksi lisätestiä, mittaamalla Wiresharkilla viestiliikennettä TLS-suojauksen ollessa käytössä. Ensimmäisessä sivutestissä haluttiin selvittää, kuinka paljon datapaketteja kulki käyttäessä suojattua porttia 8883. Testituloksista ilmeni, että datapakettien pituuksien määrä oli huomattavasti suurempi kuin ensimmäisiessä testauksissan. Viestien pituudet vaihtelivat, joista suurimmat olivat jopa 2620 tavun kokoisia. Pakettien prosentuaalinen määrä jakautui huomattavasti laajemmalle alueelle. Toisessa sivutestissä vertailtiin, millaisina suojaamattomat ja suojatut viestit näkyivät Wiresharkilla. Ensin testattiin suojaamatonta porttia 1883 ja tulokset osoittivat, että asiakkaan tunnus ja salasana näkyivät Wiresharkissa, samoin näkyi viestin sisältö. Seuraavaksi testattiin suojattua porttia 8883 ja tuloksena oli, että asiakkaan tunnus, salasana ja viestin sisältö eivät enää näkyneet Wiresharkissa. Joten salaamattoman portin kautta kulkevat viestit ovat alttiita tietoturvauhkille. Parempi vaihtoehto on lähettää viestit suojatun portin kautta.

Viidennessä testissä testattiin MQTT:n ominaisuutta viestien pysyvästä tallentamisesta. Tallentaessa viesti tallentuu tietokoneelle tietokantatiedostona. Viesti tallentui odotuksen mukaisesti Mosquitton konfiguroinnissa määriteltyyn sijaintiin. Tämä ominaisuus takaa sen, että vastaanottaja saa viestin, vaikka yhteys olisi väliaikaisesti poikki. Viesti tallentuu selkokielisenä, joten se on luettavissa ja muokattavissa esimerkiksi Notepad ++ tekstieditorilla, jolloin tiedon eheys muuttuu. Tiedoston eheydestä on mainittu teoria tekstin kappaleessa 2. Toivottavasti palvelinkoneet ovat hyvin suojattuja ja lisäsuojauksena voidaan rajoittaa kansioiden ja tiedostojen käyttöoikeuksia. MQTT:ltä löytyy vielä paljon erilaisia ominaisuuksia, joita voi testata. Työssä suoritetuilla testeillä haluttiin testata protokollan toimivuutta, käytännöllisyyttä, hyödyllisyyttä ja ennen kaikkea turvallisuutta.

5.1. MQTT ja DDS-protokollien vertailu

IoT-protokollista löytyy myös muita mielenkiintoisia protokollavaihtoehtoja, joista yksi on DDS protokolla. Eroavaisuutena MQTT:n ja DDS:n välillä on esimerkiksi DDS:stä puuttuva erillinen välittäjä. Välittäjän sijasta DDS käyttää tiedonkirjoittajia

ja datalukijoita. Lisäksi MQTT ja DDS käyttävät palvelutasoja viestin lähettämiseen, mutta DDS:llä palvelutasoja on paljon enemmän. Merkittävä eroavaisuus protokollien välillä on niitä kohtaavat palvelukatkokset. MQTT:llä on parempi resistenssi palvelukatkoksia vastaan, kuin DDS:llä. DDS:ää on käsitelty kappaleessa 3.2. Taulukko 8:ssa on koottu tarkemmin MQTT ja DDS-protokollien eroavaisuuksista.

Ominaisuudet	**MQTT**	**DDS**
Koko nimi	Message Queue Telemetry Transport	Data Distribution Service
Arkkitehtuuri	Keskitetty, viestit kulkee aina välittäjän kautta.	Hajautettu, vertaisviestintä
Kuljetuskerroksen protokolla	TCP (Transmission Control Protocol)	UDP (User Datagram Protocol)
Pääkäyttäjän tarpeellisuus	Kyllä, on asennettava Broker, kaikki asiakkaat ovat määritetty palvelimen tai välittäjän osoitteella	Ei tarvise, automaattinen etsintä (plug and play), itsemuodostuva, itseään parantava
Viestinopeus (message rate)	Muutama viesti sekunnissa laitetta kohden	Jopa 10000 viestiä sekunnissa laitetta kohden
Palvelukatkos	MQTT-verkossa siedettävä	Mahdollisesti ovat tuhoisia DDS-arkkitehtuurissa

Taulukko 8. Vertailussa kaksi tiedonsiirtoprotokollaa.[13]

13 Alkuperäinen taulukko osoitteessa https://www.rfwireless-world.com/Terminology/MQTT-vs-DDS.html.

6. POHDINTA

Diplomityön teknisen osan toteuttamisen keskeisin tehtävä oli saada vastauksia asetettuihin tutkimuskysymyksiin. Tutkimuskysymyksien teemat keskittyivät MQTT protokollan ominaisuuksiin. MQTT:n ominaisuuksia on paljon, joten tutkimuskysymykset rajoittuivat keskittymään joihinkin MQTT:n tietoturvallisuuden ominaisuuksiin. Vastauksien saaminen tutkimuskysymyksiin vaati perehtymistä aiheisiin ja kymmeniä testejä, jotta selkeitä analyyseja oli mahdollista saada. Testianalyysien vertaamiseen ei ole saatavilla täysin samanlaisia tutkimuksia. Y. Sasaki & T. Yokotani [29] ovat tutkineet työssään MQTT:n suorituskykyä, aihe sivuaa tässä työssä osaa verkkoliikenteen mittauksesta.

Ensimmäisessä testissä suoritettiin viestien lähettäminen ja vastaanottaminen Mosquitto-välittäjäohjelman avulla. Tulokset osoittivat, että viestit kulkivat onnistuneesti lähettäjältä vastaanottajalle välittäjäohjelman kautta. Testeissä testattiin myös MQTT:n eri QoS -palvelutasojen ominaisuuksia ja samalla havainnoida palvelutasojen toimivuuksien merkityksiä. QoS -palvelutasoja on tutkinut muun muassa A. Viswanathan [16]. Testeissä todettiin QoS -tasojen toimivan A. Viswanathanin esittämällä tavalla.

Tietoturvallisuuteen liittyvässä tutkimusaiheessa testattiin todennusyritysten määrää. Tulokset osoittivat, että ohjelma ei rajoittanut yrityskertoja. Oli mahdollista arvata salasana niin usein, että oikea salasana löytyi. Testistä saatiin samanlaiset tulokset, kuin G. Nebbione & M. C. Calzarossa [13] ovat esittäneet julkaisussaan.

Toisessa tietoturvallisuutta koskevassa testissä testattiin viestien lähettämistä ja vastaanottamista käyttäen TLS-suojausta. TLS-suojaus toimi hyvin ja viestit pääsivät kulkemaan suojatun portin 8883 kautta. Jatkotestissä tarkasteltiin Wiresharkilla verkkoliikenteen salauksen toimivuutta. Tuloksista kävi ilmi, että käytettäessä TLS-suojausta Wiresharkin avulla ei pystynyt näkemään viestejä selkokielisenä. TLS-ominaisuuksien testitulokset vastasivat J. Cynthia et al. [2] ja A. Viswanathanin [16] TLS:n toimintaa käsittelevissä julkaisuissa.

Tuloksien analysoinnin ja arvioinnin jälkeen voidaan olla tyytyväisiä lopputuloksiin, jotka olivat odotetun kaltaisia ja vastasivat verkkojulkaisuissa esitettyihin ominaisuuksiin. Tässä diplomityön tekeminen on antanut kokemusta oppia tuntemaan pintatasolla protokollan ominaisuuksien toimivuudesta. Erilaisten MQTT:n ominaisuuksien todentamisia testeillä rajoittaa työn aikataulu ja diplomityön tuomat rajoitteet.

MQTT:n ominaisuuksiin kuuluu, että todennusyritysten määrää ei ole rajoitettu, josta voi seurata tietoturvaongelma. Ongelmaa voisi tutkia ja etsiä ratkaisua todennusyrityksen rajaamiseksi. Toinen lisätutkimusaihe voisi olla tallennetun tietokantatiedostossa olevien viestien salaaminen.

7. YHTEENVETO

Diplomityössä tavoitteena oli tarkastella esineiden Internetissä käytettävien protokollien erilaisia ominaisuuksia. Protokollia on useita erilaisia, joten työssä keskityttiin MQTT-tiedonsiirtoprotokollan ominaisuuksiin. MQTT-protokollassa on paljon erilaisia ominaisuuksia, joten oli rajoitettava tarkastelun aluetta tietoturvallisuus ominaisuuksiin. MQTT tietoturvallisuus on diplomityön pääsisältö IoT verkkojen lisäksi.

Teknisessä osassa suoritetut testit antoivat hyviä ja selkeitä tuloksia MQTT:n ominaisuuksista, kuten asiakkaiden välillä tapahtuvasta viestien kulusta MQTT:n välittäjäohjelman kautta. Työssä testattiin viestien lähettämistä MQTT:n eri QoS-palvelutasojen välillä. Tuloksien perusteella voitiin verrata dataliikenteen määrää eri QoS-tasojen välillä. QoS 1 ja 2 -palvelutasoilla osoittautui viestien lukumäärät suuremmiksi kuin QoS 0 tasolla, johtuen Qos 1 ja 2 palvelutasojen kuittausviesteistä. Tuloksista havaittiin, että QoS 0 tason viestien perillemenoa ei varmistettu kuittausviesteillä. QoS 1 ja 2 palvelutasoilla viestien kulku on varmempaa, koska viestien perillemeno varmistetaan kuittausviesteillä.

Tietoturvallisuutta testattiin salasanan kryptauksella Mosquitto välittäjäohjelmalla. Tuloksena oli, että salasanat eivät olleet selkokielisenä palvelimen salasanatiedostossa. Salasana tiedostossa olevalla kryptatulla salasanalla ei voida kirjautua palveluun, ainoastaan selkokielinen salasana kelpaa kirjautumiseen. Salasanan kryptauksesta huolimatta kävi ilmi, että välittäjäohjelma ei rajoittanut todennusyritysten kertoja. Merkittävä havainto edellä mainituista tuloksista on, että kirjautuminen palveluun onnistuu rajoittamattoman todennusyrityksen johdosta.

Suojatun viestin kulkua testattiin käyttämällä TLS:n suojausominaisuutta. Testit osoittivat, että viestin sisältö voidaan salata ja niitä ei voida tulkita verkkoliikenteessä. Testattaessa viestin pysyvää tallennusta kävi ilmi, että viestit olivat luettavissa tietokantatiedostosta. Testien perusteella havaittiin, että TLS toimii hyvin, mutta välittäjäpalvelimelle tallentuva selkokielinen viesti voi aiheuttaa tietoturvaongelman. Ratkaisuna tietoturvaongelman korjaamiseen on suojata välittäjäpalvelimelle pääsy monivaihetunnistautumisella.

8. LÄHTEET

[1] Rousku K. (2014) Kyberturvaopas, Tietoturvaa kotona ja työpaikalla. Talentum Media Oy, Viro.

[2] Cynthia J., Parveen Sultana H., Saroja N. & Senthil J. (2019) Security Protocols for IoT. Kumaraguru College of Technology, Coimbatore, India. URL: https://www.researchgate.net/publication/328078416_Security_Protocols_for_IoT.

[3] Oracle. (luettu 8.3.2022). What is IoT. URL: https://www.oracle.com/internet-of-things/what-is-iot/.

[4] Limnéll J. & Majewski K. & Salminen M. (2014) Kyberturvallisuus. Docendo Oy, Jyväskylä.

[5] TechTarget. (luettu 8.3.2022). IoT-device. URL: https://www.techtarget.com/iotagenda/definition/IoT-device.

[6] Reijo Ratauoman säätiö. Logistiikan maailma. (luettu 17.2.2022). Esineiden Internet. URL: https://www.logistiikanmaailma.fi/logistiikka/digitalisaatio/esineiden-internet/.

[7] Empirica. (luettu 7.3.2022). Mikä on IoT. URL: https://www.empirica.fi/iot/.

[8] Centri Advanced Security for IoT. (2016). Securing the Internet of Things: 5 Key Areas to Protect. CENTRI Technology, 701 5th Ave, Suite 550, Seattle, WA 98104, USA. URL: https://www.centritechnology.com/wp-content/uploads/2019/02/CENTRI-Securing-the-IoT-White-Paper-Sept-2016.pdf.

[9] F-Secure. (luettu 18.1.2022) Kyberturvallisuus, tutkimukset & raportit, Väärennetyt laitteet tekevät yritysverkoista haavoittuvaisia. URL: https://www.f-secure.com/fi/press/p/vaarennetyt-laitteet-tekevat-yritysverkoista-haavoittuvaisia.

[10] Traficom. (luettu 18.1.2022) Kyberturvallisuuskeskus, Tietoturvan vuosi 2020, Kyberturvallisuuskeskuksen vuosikatsaus, Esineiden internet ja automaatiojärjestelmät. URL: https://www.kyberturvallisuuskeskus.fi/sites/default/files/media/publication/Tietoturvan-vuosi-2020__WEB_aukeamat.pdf.

[11] Traficom. (luettu 11.7.2022). Kybertuvallisuuskeskus, Monivaiheinen tunnistautuminen suojaa käyttäjätilejäsi. URL: https://www.kyberturvallisuuskeskus.fi/fi/ajankohtaista/ohjeet-ja-oppaat/monivaiheinen-tunnistautuminen-suojaa-kayttajatilejasi.

[12] VpnMonitor. (luettu 25.7.2022). TCP vs UDP: Understanding the Difference. URL: https://www.vpnmentor.com/blog/tcp-vs-udp/.

[13] Nebbione G. & Calzarossa M.C. (2020) Security of IoT Application Layer Protocols: Challenges and Findings. University of Pavia, Italy. URL: http://peg.unipv.it/publications/PDF/future-internet_preprint.pdf.

[14] Oasis. MQTT. (luettu 26.4.2022). MQTT Specifications. URL: https://mqtt.org/mqtt-specification/.

[15] Wallarm. (luettu 26.4.2022). What is AMQP?. URL: https://www.wallarm.com/what/what-is-amqp.

[16] Viswanathan A. (2017). Analysis of Power Consumption of the MQTT Protocol. University of Pittsburgh, Pittsburgh. URL: http://d-scholarship.pitt.edu/32399/1/Analysis%20of%20Power%20Consumption%20of%20the%20MQTT%20Protocol%20-%20Abhishek%20Viswanathan.pdf.

[17] SSL.com, SSL Support Team, (luettu 3.6.2022). Private And Public Keys. URL: https://www.ssl.com/article/private-and-public-keys/.

[18] DDS Foundation. (luettu 25.3.2022). What is DDS. URL: https://www.dds-foundation.org/what-is-dds-3/.

[19] Peniak P. & Franekova M. (2015). Open communication protocols for integration of embedded systems within Industry 4. University of Zilina, Slovak Republic. URL: https://otik.uk.zcu.cz/bitstream/11025/35117/1/Peniak.pdf.

[20] National Institute of Standards and Technology. (luettu 14.2.2022) National Vulnerability Database, CVE-2019-15136 Details. URL: https://nvd.nist.gov/vuln/detail/CVE-2019-15136.

[21] Villaverde B., Alberola R., Jara A., Fedor S., Das S. & Pesch D. (2012). Service Discovery Protocols for Constrained Machine-to-Machine Communications. United Technologies Research Centre, Cork, Ireland. URL: http://szymonfedor.eu/wp-content/uploads/2017/05/service-discovery-protocols-for-constrained-m2m-communications.pdf.

[22] Digital Guide Ionos. (luettu 24.3.2022). Multicast DNS: alternative name resolution on a small scale. URL: https://www.ionos.com/digitalguide/server/know-how/multicast-dns/.

[23] National Institute of Standards and Technology. (luettu 26.7.2022) National Vulnerability Database, CVE-2017-6520 Details. URL: https://nvd.nist.gov/vuln/detail/CVE-2017-6520.

[24] StormWall. (luettu 25.3.2022). SSDP (Simple Service Discovery Protocol). URL: https://stormwall.network/knowledge-base/protocol/ssdp .

[25] Eclipse Mosquitto. (luettu 4.2.2022). MQTT. URL: https://mosquitto.org/man/mqtt-7.html.

[26] EMQ. (luettu 11.3.2022). Introduction to MQTT QoS (Quality of Service). URL: https://www.emqx.com/en/blog/introduction-to-mqtt-qos.

[27] Wallarm. (luettu 6.4.2022). MQTT: The Enabler of Smooth And Hassle-free Information Exchange for An IoT Ecosystem. URL: https://www.wallarm.com/what/mqtt-concept.

[28] Eclipse Mosquitto. (luettu 3.2.2022). Eclipse Mosquitto An open source MQTT broker. URL: https://mosquitto.org/.

[29] Sasaki Y., Yokotani T. (2018). Performance Evaluation of MQTT as a Communication Protocol for IoT and Prototyping. Graduate School of Electrical Engineering and Electronics, Kanazawa Institute of Technology, Ishikawa, Japan. URL: https://core.ac.uk/download/pdf/228834689.pdf.