Teemu Haakana

IoT-protokollat ja niiden tietoturva

Tekniikan ja
innovaatiojohtamisen
akateeminen yksikkö
Kandidaatintutkielma
Automaatio ja tietotekniikka

Vaasa 2025

VAASAN YLIOPISTO
Tekniikan ja innovaatiojohtamisen akateeminen yksikkö
Tekijä: Teemu Haakana
Tutkielman nimi: IoT-protokollat ja niiden tietoturva
Tutkinto: Tekniikan kandidaatti
Oppiaine: Automaatio ja tietotekniikka
Työn ohjaaja: Timo Mantere
Valmistumisvuosi: 2025 **Sivumäärä:** 54

TIIVISTELMÄ:
Tietoturva on merkittävässä asemassa IoT-laitteiden sekä -järjestelmien määrän kasvaessa yhä useammalla alalla.

Tässä kandidaatintutkielmassa tarkastellaan IoT-laitteissa käytettyjen protokollien toimintaa, käyttökohteita sekä tietoturvaa. Tutkielman päätavoite on selvittää kunkin IoT-laitteissa käytetyn protokollan vahvuudet ja heikkoudet tietoturvan näkökulmasta. Tavoitteisiin kuuluu myös selvittää, miksi IoT-protokollia tarvitaan IoT-laitteiden viestinnässä ja tiedonsiirrossa.

Tutkielmassa esitellään ensin esineiden internet sekä siihen liittyviä oleellisia käsitteitä. Esineiden internet koostuu Internet-verkkoon liitetyistä antureista ja muista laitteista, jotka keräävät tietoa ympäristöstä ja välittävät tietoa keskenään. Esineiden internetin tietoturvatavoitteisiin liittyy tietojen luottamuksellisuus, eheys sekä saatavuus.

Tutkielmassa esitellään IoT:ssa tyypillisesti käytetty protokollapino, joka jakaa protokollat niiden vastuualueidensa perusteella eri kerroksiin. Protokollapino koostuu fyysisestä, siirto-, verkko-, kuljetus- sekä sovelluskerroksesta. Eri kerroksissa esiintyy erilaisia tietoturvauhkia, jotka esitellään tutkielmassa kerroskohtaisesti.

Tutkielmassa tarkastellaan protokollia MQTT, CoAP, XMPP, AMQP, LoRaWAN, Zigbee, NB-IoT ja Thread. Protokollien tarkastelussa perehdytään tietoturvan lisäksi myös niiden toimintaan ja käyttökohteisiin.

Johtopäätöksissä todetaan IoT-protokollien olevan nopeaan, tehokkaaseen ja reaaliaikaiseen tiedonsiirtoon, jolloin protokollien tietoturvaominaisuudet eivät ole niin tärkeässä asemassa. Protokollien päällä käytetään erinäisiä salausprotokollia, kuten DTLS- SSL-, ja TLS-protokollia. Salausprotokollien käyttö IoT-protokollien kanssa lisää viestinnällistä kuormitusta sekä IoT-laitteelta vaadittavaa laskentatehoa. Kaikkiin sovelluskohteisiin ei voi käyttää salausprotokollia, sillä jotkin IoT-laitteet ovat hyvin resurssirajoitteisia. Salausprotokollien välillä on eroja viestinnällisessä kuormituksessa ja IoT-laitteelta vaadittavan laskentatehon suuruudessa. IoT-protokollat valitaan sovelluskohtaisesti, jolloin voidaan valita, halutaanko IoT-järjestelmässä priorisoida enemmän esimerkiksi reaaliaikaisuutta vai tietoturvallisuutta.

AVAINSANAT: IoT, esineiden internet, tietoturva, protokollat, protokollapino

Sisällys

1 Johdanto 6

2 Internet of Things 8

3 IoT-protokollapino 10

3.1.1 Fyysinen kerros 10

3.1.2 Siirtokerros 11

3.1.3 Verkkokerros 11

3.1.4 Kuljetuskerros 11

3.1.5 Sovelluskerros 12

4 IoT-protokollat 13

4.1 MQTT 13

4.2 CoAP 17

4.3 XMPP 22

4.4 AMQP 26

4.5 LoRaWAN 29

4.6 Zigbee 32

4.7 NB-IoT 36

4.8 Thread 39

5 Johtopäätökset ja yhteenveto 43

Lähteet 49

Kuvat

Kuva 1. IoT-protokollapino, muokattu. (Sharma & Gondhi, 2018) 10
Kuva 2. Esimerkki MQTT-protokollan toiminnasta, muokattu. (HiveMQ Team, 2024b) 14
Kuva 3. TLS-kättely, muokattu (Cloudflare, n.d.-b) 16
Kuva 4. CoAP-viestintä REST-arkkitehtuurin metodeja käyttäen, muokattu (Kaskatiiski & Boyanov, 2021) 18
Kuva 5. Luotettava tiedonsiirto käyttäen CON- ja ACK-merkintöjä, muokattu (Shelby ja muut 2014) 19
Kuva 6. OSCORE-suojattu CoAP-viestintä, muokattu (Selander ja muut, 2019)..... 21
Kuva 7. Tyypillinen XMPP-viestintämalli, muokattu (Al-Fuqaha ja muut, 2015) 23
Kuva 8. XML-säkeistöistä koostuva XML-tietovirta, muokattu (Al-Fuqaha ja muut, 2015) 24
Kuva 9. LoRaWANin infrastruktuuri, muokattu (LoRa Alliance, n.d.) 30
Kuva 10. Zigbeen verkkokerroksen mesh-verkkotopologia, muokattu (Khanji ja muut, 2019)..... 34
Kuva 11. TCAT-käyttöönotto, muokattu (Thread Group, 2024) 41

Taulukot

Taulukko 1. Sovelluskerroksen protokollien ominaisuuksia 45
Taulukko 2. LoRaWAN, NB-IoT, Zigbee ja Thread ominaisuudet ja tietoturvaominaisuudet 47

Lyhenteet

AMQP Advanced Message Queuing Protocol

CoAP Constrained Application Protocol

IoT Internet of Things

LoRaWAN Long Range Wide Area Network

LPWAN Low-power wide-area network

MQTT Message Queuing Telemetry Transport

NB-IoT Narrowband IoT

SSL Secure Sockets Layer

TCP Transmission Control Protocol

TLS Transport Layer Security

UDP User Datagram Protocol

XMPP Extensible Messaging and Presence Protocol

1 Johdanto

IoT (Internet of Things) eli esineiden internet koostuu Internet-verkkoon yhdistetyistä fyysisistä laitteista ja sensoreista, jotka keräävät tietoa. IoT-laitteet välittävät tietoa keskenään ja lähettävät sitä verkkoon muodostaen infrastruktuurin, jolla laitteita voi muun muassa ohjata ja valvoa ilman ihmisen fyysistä läsnäoloa. IoT-laitteet ovat pieniä ja laskentateholtaan hyvinkin rajoitettuja, minkä takia IoT-laitteet vaativat toimiakseen erityisiä protokollia. Protokollia käytetään muun muassa tiedonsiirtoon, viestintään sekä datan muokkaamiseen tai pakkaamiseen sopivaan muotoon.

IoT-protokollat ovat yleensä hyvin kevyitä ja niissä on tietoturvaongelmia, jotka johtuvat pääosin IoT-laitteiden rajallisista resursseista. Protokollat priorisoivat tehokasta ja nopeaa viestintää sekä tiedonsiirtoa tietoturvan sijaan. Osaa protokollista on käytetty jo useamman vuoden ajan IoT-laitteissa, eikä niissä ole huomioitu nykypäivän tietoturvavaatimuksia. Tietoturva on merkittävässä asemassa IoT-laitteiden määrän kasvaessa yhä useammalla alalla ja niiden käsittelemän arkaluontoisen datan lisääntyessä. IoT-laitteiden tietoturvaongelmat voivat esimerkiksi aiheuttaa ongelmia kriittisten infrastruktuurien kuten sähköverkkojen toiminnassa. Tämän lisäksi yksityisyyden suoja voi tulla loukatuksi IoT-laitteiden käsittelemien tietojen joutuessa vääriin käsiin.

Tutkielman tarkoituksena on esitellä kirjallisuuskatsauksen avulla IoT-laitteissa yleisimmin käytetyt protokollat sekä niiden toiminta, ja selvittää, mihin käyttötarkoitukseen ne ovat, ja miksi niitä tarvitaan IoT-infrastruktuurissa. Tutkielmassa pyritään selvittämään kunkin protokollan heikkoudet ja vahvuudet tietoturvan näkökulmasta. Olen valinnut tutkielmaan IoT-laitteissa yleisimmin käytetyt sovelluskerroksen IoT-protokollat MQTT, CoAP, XMPP sekä AMQP. Edellä mainittujen lisäksi tarkastellaan siirto- ja verkkokerroksen protokollia LoRaWAN, Zigbee ja NB-IoT. Tarkastelun kohteeksi valitsin myös IoT-laitteisiin erikoistuneen Thread-protokollan. Protokollat on valittu työhön niiden yleisyyden perusteella.

Luvussa 2 esitellään lyhyesti IoT eli esineiden internet (engl. Internet of Things), ja siihen oleellisesti liittyvät käsitteet sekä eri osa-alueet. Luvussa 3 esitellään IoT-protokollapino, jonka avulla voidaan hahmottaa protokollien eri tehtävät ja protokollien tietoturviin liittyvät ongelmat. Luvussa 4 esitellään työssä käsiteltävät protokollat, niiden toiminta sekä niiden tietoturvaominaisuudet ja mahdolliset tietoturvahaavoittuvuudet- ja uhat. Luvussa 5 tehdään yhteenveto työstä, ja käydään läpi työssä tehtyjä johtopäätöksiä IoT-protokollista ja niiden tietoturvasta.

2 Internet of Things

Esineiden internet eli IoT (engl. Internet of Things) on esitelty ensimmäisen kerran käsitteenä jo vuonna 1999 Kevin Ashtonin käyttäessä termiä ”Internet of Things” Procter & Gamble -yhtiölle tehdyssä esitelmässään radiotaajuisen etätunnistuksen liittämisestä toimitusketjuun (Ashton, 2009). Esineiden internetin esineet ovat Internet-verkkoon yhdistettyjä laitteita ja antureita, joiden tehtäviin kuuluu muun muassa tietojen kerääminen ympäristöstään. Anturit keräävät tietoa esimerkiksi lämpötilasta, ilmankosteudesta ja sijainnista. Esineet jakavat keräämiään tietoja keskenään ja analysoivat tietoja reaaliaikaisesti. Esineiden keräämien ja jakamien tietojen avulla esineet voivat suorittaa tehtäviä ja tehdä päätöksiä automaattisesti ilman ihmisen fyysistä läsnäoloa. Esineet muodostavat älykkäitä järjestelmiä hyödyntäen viestintäteknologioita, antureiden muodostamia verkkoja, sekä Internet-protokollia ja -sovelluksia. (Al Fuqaha ja muut, 2015)

Internet of Thingsin sovelluskohteisiin kuuluvat muun muassa liikenne, terveydenhuolto teollinen automaatio sekä älykkäät kodit. IoT mahdollistaa merkittävässä asemassa olevia elämänlaatua parantavia älykkäitä järjestelmiä, joiden määrä on huomattavassa kasvussa. IoT-pohjaiset älykkäät järjestelmät edistävät maailman talouskasvua, ja niiden taloudellinen kasvu on merkittävää liiketoiminnassa, etenkin terveydenhuollossa ja valmistusteollisuudessa. (Al Fuqaha ja muut, 2015)

Tietoturvan osalta IoT:n tavoitteisiin kuuluu tietojen luottamuksellisuuden (engl. data confidentiality), tietojen eheyden (engl. data integrity) ja tietojen saatavuuden (engl. data availability) toteutuminen. Tietojen luottamuksellisuudella tarkoitetaan yksityisyyden takaamista käyttäjälle muodostamalla turvallinen yhteys vain sallituille käyttäjille. Tietojen luottamuksellisuus toteutuu käyttämällä salausmekanismia, jossa jokainen tietojen sisältämä bitti muunnetaan salatekstiksi, jota seuraa kaksivaiheinen varmennusprosessi. Varmennusprosessissa kaksi laitetta tai muuta IoT-komponenttia suorittavat todennustestin. Pääsy tietoihin myönnetään vain molempien laitteiden läpäistyä todennustestin sekä biometrisen vahvistuksen, jonka avulla tietoja pyytänyt

henkilö on yksilöitävissä. Tietojen eheys on integroitu verkkoon, jotta tiedot saadaan suojattua ilman, että kyberrikolliset voivat muokata niitä järjestelmän huomaamatta tietojen väärinkäyttöä. Tietojen eheys tarkistetaan käyttäen erinäisiä virheenhavaitsemismenetelmiä ja tietoja synkronoidaan jatkuvasti varmuuskopiointitarkoituksia varten. Tietojen saatavuus on IoT:n tärkein tietoturvatavoite. Tietojen saatavuudella tarkoitetaan sitä, että käyttäjille välitetään IoT-laitteiden keräämiä tietoja silloin, kun niitä tarvitaan. Tietojen saatavuuden tulisi toteutua myös normaaleista olosuhteista poikkeavissa tilanteissa. (Swamy ja muut, 2017) Tietojen luottamuksellisuuden, eheyden ja saatavuuden merkitys korostuu IoT-laitteiden määrän kasvaessa miljardeihin ympäri maailmaa (McAteer ja muut, 2017).

Granjal ja muut ovat todenneet Rahmanin ja Shanin (2016) työssä esineiden internetin tietoturvan olevan tärkeä aihe, koska IoT-järjestelmät käsittelevät arkaluonteisia tietoja internetin välityksellä. Näin ollen tietoturva, yksityisyys ja luottamus ovat keskiössä IoT-järjestelmien toteuttamisessa. Heidän mukaansa suurimmiksi haasteiksi koituvat suorituskyky ja nopeus, mikäli tietoturvaominaisuuksia lisätään IoT-järjestelmiin. IoT-laitteet ovat kevyitä ja niiden valmistuksessa on usein painotettu niiden vähäistä ja rajoitettua laskentatehoa sekä suurta muistikapasiteettia. Tämä mahdollistaa viiveen minimoimisen käyttäjien viestinnässä ja viestien perille menemisen sellaisissa ympäristöissä, joissa tapahtuu pakettihävikkiä.

3 IoT-protokollapino

IoT-protokollat ja standardit voidaan jakaa niiden kommunikointiin ja tiedonsiirtoon liittyvien vastuualueidensa perusteella IoT-protokollapinon eri kerroksiin OSI-mallin tavoin. IoT-protokollapino määrittää IoT-laitteiden tiedonsiirron ja kommunikoinnin laitteiden välillä ja Internet-verkon yli. Yksi tapa esittää IoT-protokollapino on jakaa protokollat ja standardit viiteen eri kerrokseen (Kuva 1), jotka ovat: fyysinen kerros (physical layer), siirtokerros (data-link layer), verkkokerros (network layer), kuljetuskerros (transport layer), sekä sovelluskerros (application layer) (Sharma & Gondhi, 2018).

Kuva 1. IoT-protokollapino, muokattu. (Sharma & Gondhi, 2018)

3.1.1 Fyysinen kerros

Fyysinen kerros määrittää, kuinka kukin laite on fyysisesti liitetty verkkoon, mikä voi olla toteutettu käyttäen esimerkiksi optisia kaapeleita, johtoja tai radioaaltoja langattomissa verkoissa, kuten WiFi 802.11 a/b/g/n (Gerber & Romeo, 2020). Fyysinen kerros on IoT-protokollapinon alin kerros ja siihen kuuluvat muun muassa standardit IEEE 802.15.4 ja

IEEE 802.15.4e (Sharma & Gondhi, 2018). Fyysisen kerroksen tietoturvauhkiin kuuluu signaalin häirintä, salakuuntelu, laitteiston manipulointi sekä hakkerointi (Tomić & McCann, 2017).

3.1.2 Siirtokerros

Siirtokerroksen protokollien tärkeimpiin tehtäviin kuuluu tietojen siirtäminen fyysisen kerroksen yli. Siirtokerros tarjoaa selkeästi määritellyn rajapinnan IoT-protokollapinon ylemmälle kerrokselle eli verkkokerrokselle. Siirtokerros vastaa tiedonsiirrossa tapahtuvien virheiden hallinnasta, huolehtii alkuperäisestä yhteyden muodostamisesta sekä sopivan fyysisen protokollan määrittämisestä tiedolle. Protokollat, kuten Bluetooth, BLE ja Z-Wave toimivat IoT-protokollapinon siirtokerroksessa. (Sharma & Gondhi, 2018)

Siirtokerroksessa tietoturvaa uhkaaviin hyökkäyksiä tapahtuu yleensä siirtokerroksen MAC-alakerroksessa (Medium Access Control), ja näitä hyökkäyksiä ovat signaalin häirintä sekä törmäykset (engl. collisions) (Tomić & McCann, 2017).

3.1.3 Verkkokerros

Verkkokerroksen tehtäviin kuuluu reitittäminen, jonka tehtävä on siirtää paketit niiden lähteestä niiden määränpäähän, sekä kapselointi, joka vastaa tietojen muuttamisesta lähetettäviksi paketeiksi (Alahari & Yalavarthi, 2017). Verkkokerroksessa käytettäviin protokolliin kuuluu IPv4- ja IPv6 -IP-protokollat, sekä vähävirtaisiin langattomiin verkkoihin käytettäväksi suunniteltu 6LowPAN (IPv6 over Low power Wireless Personal Area Networks), joka mahdollistaa kommunikoinnin käyttäen IPv6-protokollaa (Al-Fuqaha ja muut, 2015).

3.1.4 Kuljetuskerros

Kuljetuskerros vastaa luotettavasta tietojen siirrosta (Tomić & McCann, 2017). Kuljetuskerroksessa käytettäviin protokolliin kuuluu TCP- sekä UDP-protokollat (Sharma & Gondhi, 2018). TCP (Transmission Control Protocol) on laajasti käytetty

kuljetuskerroksen protokolla, joka varmistaa luotettavan, järjestetyn sekä virhetarkistetun tiedonsiirron verkossa toimivien sovellusten välille (Postel, 1981). UDP (User Datagram Protocol) on yhteydetön kuljetuskerroksen protokolla, jota käytetään kommunikointiin matalaa viestinnällistä kuormitusta suosivien sovellusten välillä. Toisin kuin TCP, UDP ei takaa luotettavaa, järjestettyä ja virhetarkistettua tiedonsiirtoa. UDP-protokolla priorisoi viestinnän nopeutta luotettavuuden sijasta. (Postel, 1980)

3.1.5 Sovelluskerros

Sovelluskerroksen tehtäviin kuuluu tarjota käyttäjille niiden pyytämiä palveluita, jotka vastaavat käyttäjien tarpeita. Sovelluskerroksen avulla käyttäjille voidaan välittää tietoa esimerkiksi IoT-sensoreiden keräämistä lämpötilamittauksista käyttäjien pyytäessä niitä. Sovelluskerros kattaa useita IoT:n sovelluskohteita, kuten älykkäät kodit, teollisen automaation, terveydenhuollon sekä liikenteen. (Al-Fuqaha ja muut, 2015)

Sovelluskerroksen pääasialliset tietoturvaongelmat liittyvät haavoittuvuuksiin, joita voidaan hyödyntää haittaohjelmahyökkäyksillä. Hyökkäykset voivat vaikuttaa laitteiston ja laitteiston ohjelmiston toimintaan. (Tomić & McCann, 2017)

4 IoT-protokollat

Tietoliikenteessä protokollalla tarkoitetaan standardisoituja sääntöjä, joita käytetään tietojen käsittelyyn sekä niiden muotoiluun (Cloudflare, n.d.-a). IoT-laitteet noudattavat erilaisia protokollia mahdollistaakseen kommunikoinnin ja tiedonsiirron sekä keskenään, että Internet- tai mobiiliverkon yli. IoT-laitteissa käytetyissä protokollissa tulee ottaa huomioon laitteiden rajalliset resurssit sekä laitteiden toimintaympäristö. Tavalliset Internet-protokollat eivät tarjoa riittävää suorituskykyä eikä tehokkuutta IoT-laitteiden kommunikointiin tai tiedonsiirtoon. Näin ollen IoT-laitteet tarvitsevat erityisiä protokollia kommunikoinnin toteuttamiseksi tehokkaasti. Vaikka IoT-esineiden käsittelemät tietomäärät saattavat olla suhteellisen pieniä, laitteiden käyttämä energia ja kaistanleveys kasvaa merkittävästi IoT-laitteiden määrän kasvaessa. On tärkeää valita sopivat IoT-protokollat sovelluskohtaisesti tehokkaan kommunikoinnin takaamiseksi ja laitteiden suorituskyvyn sekä kustannustehokkuuden parantamiseksi. (Kaskatiiski & Boyanov, 2021)

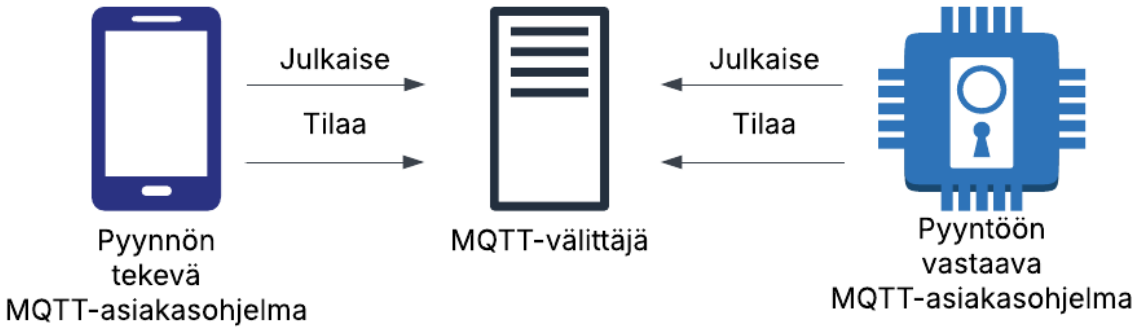
4.1 MQTT

MQTT (Message Queuing Telemetry Transport) on vuonna 1999 julkaistu IoT-laitteissa yleisesti käytetty sovelluskerroksen protokolla. MQTT-protokollan ensimmäinen versio suunniteltiin alun perin öljyputkien seuraamiseen. MQTT:n keksijät Andy Stanford-Clark ja Arlen Nipper asettivat useita vaatimuksia protokollalle, joihin kuului yksinkertainen toteutus, tietoliikenteen luokittelua ja priorisointia tukeva tietojen toimitus (Quality of Service), keveys, kaistanleveyden säästäminen, riippumattomuus tiedoista ja jatkuva istuntojen hallinta. Vaatimukset ovat edelleen tärkeässä osassa MQTT-protokollan ominaisuuksien osalta. OASIS standardisoi protokollan vuonna 2014. (HiveMQ Team, 2024b)

MQTT-protokollalla on monta eri käyttökohdetta IoT:ssä, teollisessa IoT:ssä sekä koneiden välisessä viestinnässä. Sitä käytetään älykkäissä kodeissa muun muassa termostaateissa ja turvakameroissa, joita voi ohjata langattomasti mobiilisovelluksen

avulla. MQTT-protokollaa käytetään teollisuusautomaatiossa koneiden ja sensorien yhdistämiseksi, mikä mahdollistaa teollisten prosessien reaaliaikaisen seurannan parantaen prosessien tehokkuutta. MQTT on käytössä myös maanviljelyssä maan kosteuden ja olosuhteiden seuraamiseksi, terveydenhuollossa potilaiden etäseurannan toteuttamiseksi, sekä liikenteessä ajoneuvojen seuraamiseksi turvallisuuden parantamiseksi. (HiveMQ Team, 2024b)

MQTT-protokollan tiedonsiirron osapuoliin kuuluu tilaaja (engl. subscriber), julkaisija (engl. publisher) sekä välittäjä (engl. broker). Tilaajia ja julkaisijoita kutsutaan asiakasohjelmiksi (engl. client). Protokolla noudattaa IoT-protokollille tyypillistä julkaisija/tilaaja -mallia, jossa julkaisija lähettää aiheen eli viestin välittäjälle. Viestit kulkevat välittäjän kautta tilaajille, jotka ovat tilanneet kyseisen aiheen välitettäväksi itselleen. Välittäjäpalvelin on vastuussa tilausten seuraamisesta sekä aiheiden välittämisestä kyseisen aiheen tilaajille. Kuvassa 2 on havainnollistettu protokollan toimintaa oven avaamiseen älykkäässä kodissa mobiilisovellusta käyttäen. (HiveMQ Team, 2024b)



Kuva 2. Esimerkki MQTT-protokollan toiminnasta, muokattu. (HiveMQ Team, 2024b)

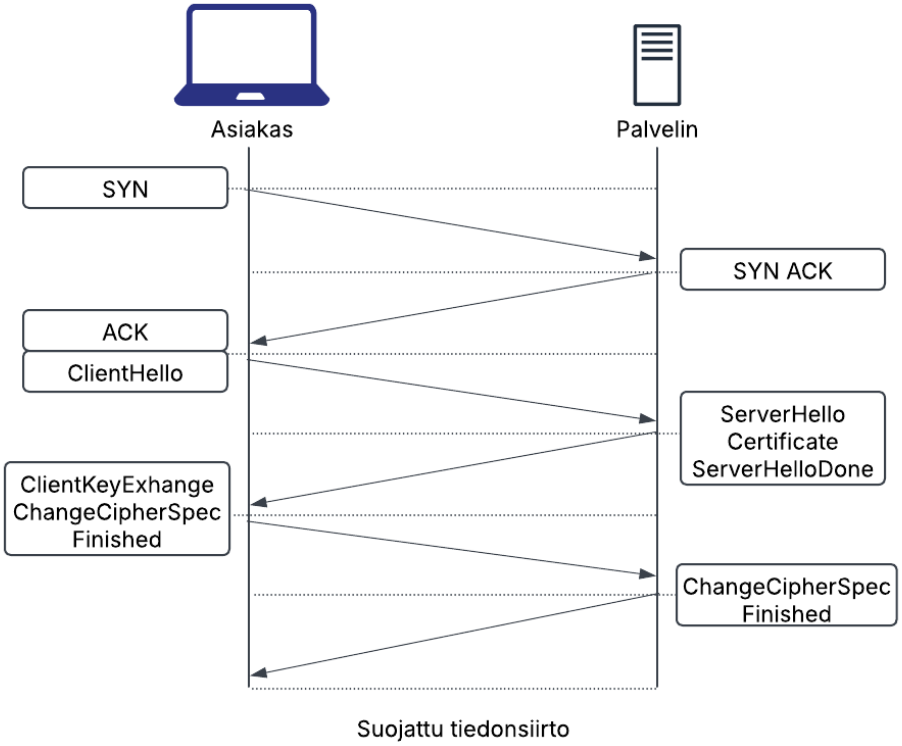
Kuvassa 2 pyynnön tekevä MQTT-asiakasohjelma julkaisee pyynnön oven avaamisesta ja tilaa tiedon oven avaamisesta pyyntöön vastaavalta asiakasohjelmalta. Pyyntöön vastaava MQTT-asiakasohjelma tilaa pyynnön ja julkaisee tiedon oven avaamisesta.

MQTT on yksinkertainen tehokasta viestintää priorisoiva protokolla, minkä vuoksi MQTT-protokollan käytössä ilmenee tietoturvaongelmia. Suurin osa tietoturvaongelmista liittyy tilaan, jossa MQTT toimii oletusarvoisesti, ja tietoturvaongelmat voidaan korjata konfiguroimalla MQTT oikein sekä käyttämällä salausprotokollia. Oletusarvoisesti MQTT-protokollassa ei ole käytössä turvallista todennusmenetelmää, mikä saattaa johtaa viestinnän osapuolten identiteettien väärentämiseen tai luvattomien tietojen siirtoon. Todennukseen liittyvä tietoturvaongelma voidaan ratkaista konfiguroimalla protokollan asetukset niin, että asiakasohjelmat välittävät käyttäjätunnuksen ja salasanan CONNECT-viestin mukana muodostaessaan yhteyden MQTT-välittäjään käyttäjän todentamiseksi. Todennusmenetelmän puuttumisen lisäksi myös käyttöoikeuksien hallinta puuttuu oletusarvoisesti. Todennetut asiakasohjelmat voivat julkaista ja tilata aiheita, joihin niillä ei ole käyttöoikeuksia, mikä on tietoturvallisesti merkittävä ongelma. Käyttöoikeuksien hallinta tulee toteuttaa MQTT-välittäjän asetuksissa asettamalla aihekohtaiset käyttöoikeudet. (Hernández Ramos ja muut, 2018)

MQTT kohtaa tietoturvaongelmia myös tietojen luottamuksellisuuden ja eheyden osalta. Oletusarvoisesti MQTT käyttää viestintään kuljetuskerroksessa TCP-protokollaa, jolloin viestintää ei ole salattu, ja samassa verkossa olevat käyttäjät voivat kuunnella verkossa tapahtuvaa viestintää. Viestinnän salaamiseksi tulisi käyttää TLS-suojausta TCP:n päällä pelkän TCP-protokollan sijasta. Tietojen eheyden varmistamiseksi MQTT tukee tarkistussummaa, viestin todentamiskoodia (MAC) sekä digitaalisia allekirjoituksia, jotka ovat menetelmiä, joilla varmistetaan viestinnässä vaihdettujen pakettien eheys. Tietojen eheys on tärkeää etenkin TLS-suojauksen ollessa pois käytöstä ja epäluotettavilla tai tunnistamattomilla asiakasohjelmilla ollessa pääsy MQTT-välittäjäpalvelimeen sekä aiheisiin. (Hernández Ramos ja muut, 2018)

Turvallinen yhteys asiakasohjelman ja välittäjäpalvelimen välille muodostetaan TLS- ja SSL-salausprotokollia käyttäen (HiveMQ Team, 2024a). TLS (Transport Layer Security) on salausprotokolla, jonka avulla voidaan salata viestinnässä vaihdetut tiedot kolmansilta osapuolilta, todentaa käyttäjät sekä varmistaa tietojen eheys asiakasohjelman ja

palvelimen suoritettua TLS-kättelyn (Cloudflare, n.d.-b). SSL (Secure Sockets Layer) on vuonna 1995 julkaistu salausprotokolla, joka nimi päivittyi myöhemmin TLS-protokollaan omistajanvaihdoksen seurauksena pääasiallisten protokollan ominaisuuksien pysyessä hyvin samanlaisina (Cloudflare, n.d.-c). TLS-kättelyssä määritellään, mitä TLS-versiota ja salausmenetelmää käytetään. Tämän lisäksi palvelin varmennetaan palvelimen TLS-sertifikaatilla, ja luodaan istuntoavaimet viestien salaamiseksi TLS-kättelyn päätyttyä. (Cloudflare, n.d.-b) Kuvassa 3 on esitetty TLS-kättelyn eri vaiheet.



Kuva 3. TLS-kättely, muokattu (Cloudflare, n.d.-b)

MQTT-protokollaa käyttäessä TLS-salausprotokollaa tulisi käyttää lähtökohtaisesti aina, kun mahdollista. Vaikka TLS lisää MQTT:n tietoturvaa, sen käyttäminen tuo mukanaan myös haittapuolen: sen käyttäminen vaatii enemmän laskentatehoa ja lisää viestinnällistä kuormitusta. Uuden TLS-yhteyden muodostaminen vaatii jopa muutaman kilotavun kaistanleveyttä. TLS voi olla ongelmallinen etenkin hyvin resurssirajoitteisissa

laitteissa, joita ei ole suunniteltu paljon laskentatehoa vaativien tehtävien suorittamiseen. (HiveMQ Team, 2024a)

Yksi tapa vähentää viestinnällistä kuormitusta on istunnon jatkaminen (engl. session resumption), jossa voidaan uudelleenkäyttää aiemmin luotua TLS-istuntoa yhteyden muodostamisen jälkeen. Tällöin asiakasohjelman ja palvelimen ei tarvitse suorittaa laskentatehoa vaativaa ja viestinnällistä kuormitusta lisäävää TLS-kättelyä uudestaan salatun yhteyden muodostamiseksi. Mikäli TLS ei sovi käytettäväksi laitteiden resurssien ollessa vähäisiä, voi tietoturvaa lisätä myös salaamalla pelkästään julkaistavien aiheiden tietosisällön sekä salasanan välittäjäpalvelimelle lähetettävässä CONNECT-viestissä asiakasohjelman muodostaessa yhteyttä välittäjäpalvelimeen. (HiveMQ Team, 2024a)

4.2 CoAP

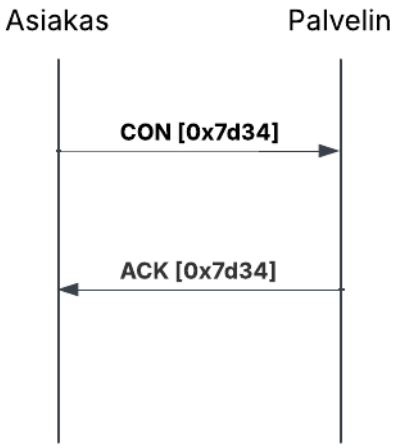
CoAP (Constrained Application Protocol) on sovelluskerroksen protokolla, joka on suunniteltu käytettäväksi tehokkaaseen viestintään resurssirajoitetuissa laitteissa sekä verkoissa. CoAP-protokollaa käytetään usein koneiden välisessä viestinnässä eli M2M-sovelluksissa (Machine to Machine). CoAP toimii kuljetuskerroksen UDP-protokollan päällä, ja se on ominaisuuksiltaan kevyt sekä matalan viestinnällisen kuormituksen protokolla. CoAP soveltuu käytettäväksi yhdessä HTTP-protokollan kanssa mahdollistaen integroinnin erinäisten web-sovellusten kanssa. (Shelby ja muut, 2014)

CoAP:n pääasiallisiin käyttökohteisiin kuuluu sovellukset älykkäissä energiajärjestelmissä, rakennusautomaatiossa sekä koneiden välisessä viestinnässä. CoAP-protokollaa käyttävät laitteet, joita kutsutaan solmuiksi (engl. node), sisältävät usein 8-bittisen mikro-ohjaimen, joka on rajoitettu niin ROM- kuin RAM-muistin osalta. Laitteet eli solmut liitetään rajoitettuihin verkkoihin kuten esimerkiksi IPv6-verkkoon. Rajoitetut verkot ovat tiedonsiirtonopeuksiltaan kykeneviä siirtämään kymmeniä kilobittejä sekunnissa, ja verkkojen käsitellessä paketteja tapahtuu usein virheitä. CoAP on suunniteltu ja optimoitu toimimaan edellä mainituissa rajoitetuissa laitteissa ja verkoissa mahdollistaen tehokkaan viestinnän. (Shelby ja muut, 2014)

Shelbyn ja muiden (2014) standardissa todetaan CoAP-protokollan noudattavan niin sanottua pyyntö/vastaus- mallia, jossa asiakasohjelma tekee pyyntöjä palvelimella sijaitsevaan resurssiin tai jonkin toiminnon suorittamiseksi. Tyypillisessä CoAP-toteutuksessa CoAP-nodet eli solmut toimivat samanaikaisesti sekä asiakasohjelmina että käyttäjinä, sillä viestintä on tyypillisesti koneiden välistä. CoAP voidaan rinnastaa HTTP-protokollaan, koska CoAP käyttää samaa REST-arkkitehtuuria kuin HTTP. CoAP-protokolla käyttää viestinnässään REST-arkkitehtuurin metodeja GET, POST, PUT ja DELETE. CoAP-ympäristössä metodeja käytetään pyynnöissä palvelimella sijaitsevan resurssin hakemiseksi tai muokkaamiseksi. Kuvassa 4 on esitetty viestintä käyttäen REST-metodeja.

Kuva 4. CoAP-viestintä REST-arkkitehtuurin metodeja käyttäen, muokattu (Kaskatiiski & Boyanov, 2021)

Viestinnässään CoAP käyttää neljää eri viestityyppiä, joilla vaikutetaan viestinnän luotettavuuteen. Tärkeät viestit merkitään Confirmable (CON) -viesteiksi. Palvelimelle pyynnön tekevä asiakasohjelma lisää Confirmable-merkinnän lähettämäänsä pyyntöön, johon palvelin vastaa Acknowledgement (ACK) -viestillä, jolla palvelin varmistaa saaneensa pyynnön asiakasohjelmalta. (Shelby ja muut, 2014) Luotettava tiedonsiirto CoAP-laitteen ja palvelimen välillä tapahtuu kuvan 5 mukaisesti.



Kuva 5. Luotettava tiedonsiirto käyttäen CON- ja ACK-merkintöjä, muokattu (Shelby ja muut 2014)

Non-Confirmable (NON) -viesteiksi merkitään sellaiset viestit, jotka eivät vaadi vahvistusta eli ACK-viestiä palvelimelta. Tällaisia viestejä voivat olla esimerkiksi yksittäiset sensorin tekemät mittaukset lämpötilasta, joiden menetys ei ole kriittistä. NON-viesteihin voidaan vastata RESET-viestillä, jos palvelin ei pysty käsittelemään asiakasohjelman tekemää pyyntöä tai palvelin ilmoittaa asiakasohjelmalle virhetilanteesta. (Shelby ja muut, 2014)

CoAP ei oletusarvoisesti tarjoa tietoturvaominaisuuksia, kuten tietojen luottamuksellisuutta tai käyttäjän todennusta, vaan tietoturvaa joudutaan lisäämään salausprotokollilla. CoAP:n kanssa yhdessä käytettäviin salausprotokolliin kuuluu DTLS (Datagram Transport Layer Security) sekä OSCORE (Object Security for Constrained RESTful Environments). DTLS on kuljetuskerroksen UDP-protokollan päällä kulkeva TLS-salausprotokollaan perustuva salausprotokolla tietoliikenteen salaamiseksi, jossa on vain pieniä muutoksia TLS-protokollaan verrattuna. OSCORE julkaistiin vuonna 2019 DTLS-salausprotokollaa kevyemmäksi vaihtoehdoksi. (Suleymanov ja muut, 2022)

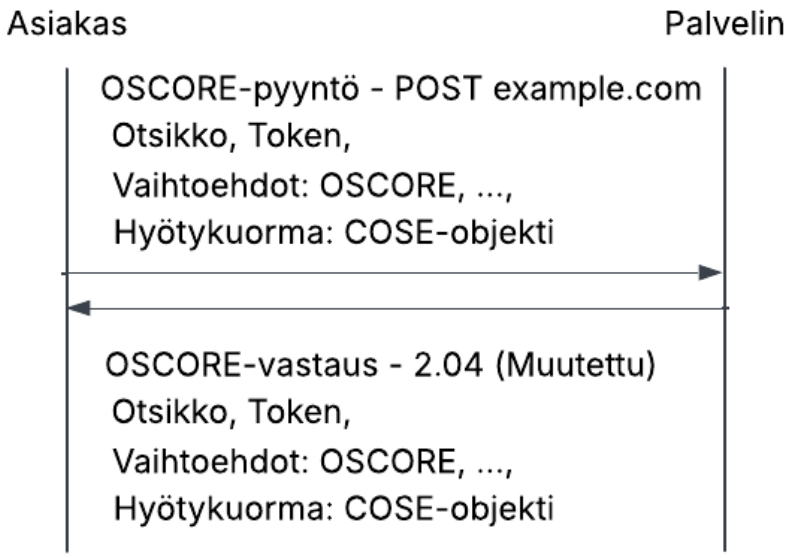
CoAP-ympäristössä tiedonsiirto on asynkronista, ja se toteutetaan datagrammeihin perustuvan protokollan, usein UDP-protokollan avulla (Shelby ja muut, 2014). DTLS-salausprotokolla on suunniteltu salaamaan datagrammeihin perustuva tiedonsiirto. Datagrammeja hyödyntävässä tiedonsiirrossa TLS-suojausta ei voi käyttää, sillä paketit voivat kadota tai saapua väärässä järjestyksessä. TLS-suojausta ei ole suunniteltu käsittelemään tällaista epäluotettavuutta tiedonsiirrossa, sillä TLS muun muassa olettaa TLS-kättelyn muodostamisen tiedonsiirron salaamiseksi tapahtuvan luotettavasti ilman, että viestejä katoaa. DTLS-suojaus korjaa datagrammeihin perustuvaan tiedonsiirtoon liittyvään epäluotettavuuteen liittyvät ongelmat pitäen samalla TLS-suojauksen perusperiaatteet. (Rescorla & Modadugu, 2012)

Shelbyn ja muiden (2014) standardissa CoAP-protokollalle on määritelty neljä eri suojaustilaa, jotka ovat NoSec, PreSharedKey, RawPublicKey ja Certificate. Suojaustilat tarjoavat eri tasoja DTLS-salausprotokollan käytölle CoAP-protokollaa käytettäessä. NoSec-tilassa DTLS-suojaus ei ole ollenkaan käytössä. NoSec-tilassa tieto yksinkertaisesti lähetetään salaamattomina paketteina UDP- ja IP-protokollien päällä. PreSharedKey-suojaustilassa DTLS-suojaus on käytössä. PreSharedKey-tilassa on jaettu lista avaimia, joista kukin sisältää listan laitteista, joiden kanssa kyseistä avainta voi käyttää viestinnän mahdollistamiseksi. RawPublicKey-suojaustilassa DTLS-suojaus on päällä, ja laitteet käyttävät julkisia avaimia ilman X.509-varmenteita. Kullekin laitteelle on laskettu identiteetti julkisesta avaimesta, ja laitteilla on myös lista muista identiteeteistä, joiden kanssa viestintä on mahdollista. Certificate-suojaustilassa käytetään DTLS-suojausta ja X.509-varmenteita avaimissa.

Kaikki DTLS-suojaustilat ei sovellu aina käytettäväksi CoAP:n kanssa, mikäli CoAP-solmut ovat muistiltaan hyvin rajoittuneita, tai vastaavasti käytössä olevien verkkojen kaistanleveyttä on rajoitettu. Osa suojaustilojen sisältämistä salausalgoritmeista lisää CoAP-toteutuksen monimutkaisuutta, minkä lisäksi DTLS-kättelyn viestinnällinen kuormitus lisääntyy 13:lla tavulla yhtä datagrammia kohden. DTLS-suojauksen

soveltuvuutta tulee harkita sovelluskohtaisesti ottaen huomioon CoAP-toteutuksen saatavilla olevat resurssit ja salausalgoritmit. (Shelby ja muut, 2014)

OSCORE (Object Security for Constrained RESTful environments) -salausprotokolla mahdollistaa päätepisteiden välisen viestinnän suojaamisen päästä päähän sellaisissa CoAP-sovelluksissa, joissa CoAP-solmut ja -verkot ovat resursseiltaan hyvin rajoittuneita. OSCORE-salausprotokolla toimii joko TCP- tai UDP-kuljetusprotokollan päällä. Protokolla käyttää viestien salaamiseen esijaettuja avaimia. CoAP-viestit muunnetaan OSCORE-viesteiksi, jotka sisältävät alkuperäisen viestin salattuna COSE (CBOR Object Signing and Endcryption) -objektissa. COSE-objekti välitetään vastaanottajalle CoAP-viestin hyötykuormassa eli payloadissa. Viestin vastaanottaja purkaa COSE-objektin salauksen ja muuntaa sen takaisin alkuperäiseksi viestiksi, sekä varmistaa viestin aitouden. (Selander ja muut, 2019) Kuvassa 6 on esimerkki OSCORE-suojatusta CoAP-viestinnästä, jossa asiakas lähettää POST-pyynnön palvelimelle. Palvelin vahvistaa asiakkaan tekemän pyynnön onnistuneen käsittelyn 2.04-tilakoodilla.



Kuva 6. OSCORE-suojattu CoAP-viestintä, muokattu (Selander ja muut, 2019)

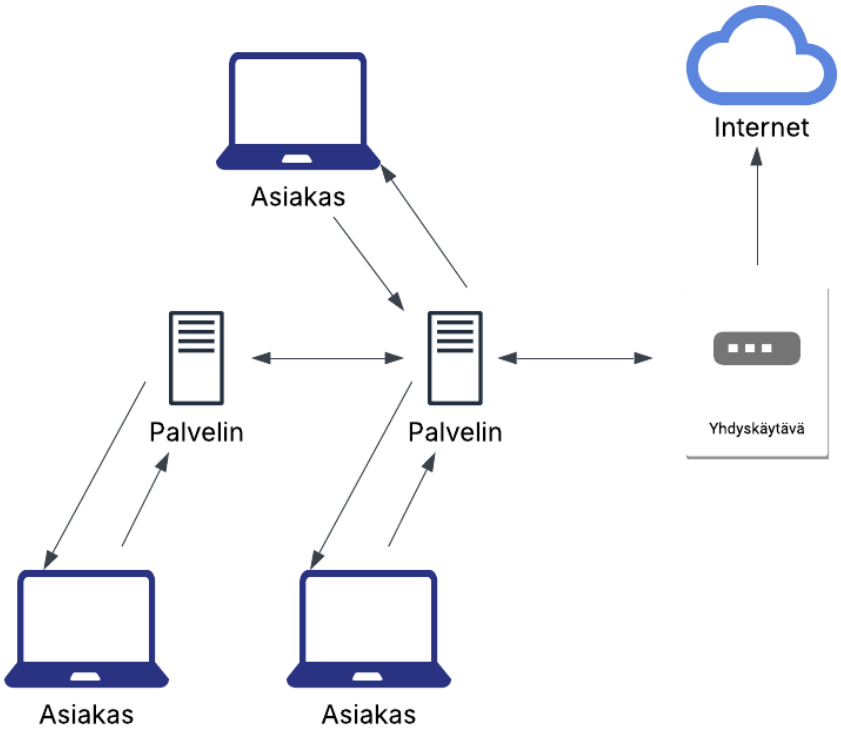
Suleymanov ja muut ovat vertailleet työssään (2022) DTLS- ja OSCORE protokollien soveltuvuutta CoAP-viestinnän suojaamiseen. OSCORE-protokollaa käytettäessä viestinnällinen kuormitus on paljon pienempi verrattuna DTLS-protokollaan, mutta toisaalta OSCORE ei salaa koko viestiä jättäen salaamatta osan viestin sisältämistä otsikkokentistä. Tämä heikentää viestinnän tietoturvallisuutta, mutta säästää laskennallisia resursseja sekä kaistanleveyttä. DTLS-salausprotokollaa käytettäessä päästä päähän -suojaus ei ole mahdollista, mikäli CoAP-sovelluksen verkoissa käytetään välityspalvelimia tai yhdyskäytäviä. Työssä todettiin salausprotokollan valinnan riippuvan siitä, kuinka paljon ja kuinka nopeasti viestejä lähetetään tietyssä ajassa. DTLS on parempi pitkäkestoisissa istunnoissa, joissa viestintää tapahtuu paljon, kun taas OSCORE suoriutuu paremmin viestinnän ollessa vähäistä.

4.3 XMPP

XMPP (Extensible Messaging and Presence Protocol), aikaisemmalta nimeltään Jabber, on avoimen standardin alun perin nopeaan XML-pohjaiseen viestintään suunniteltu sovelluskerroksen protokolla, joka julkaistiin ensimmäisen kerran vuonna 1999 (XMPP, n.d.-a). XML-kieli on tietojen tallentamiseen ja siirtämiseen kehitetty merkintäkieli, jossa tietoja merkitään erilaisten tunnisteiden avulla (W3Schools, n.d.). XMPP standardisoitiin virallisesti vuonna 2004 Internet Engineering Task Forcen toimesta. XMPP on vuosien saatossa laajentunut useampiin eri käyttökohteisiin, joihin kuuluu muun muassa läsnäolotietojen seuranta, XML-datan reititys ja kevyenä välitysohjelmana (engl. middleware) toimiminen. (XMPP, n.d.-a)

XMPP-protokolla soveltuu erinomaisesti käytettäväksi IoT-sovelluksissa sen avoimen lähdekoodin ja laajan käytön ansiosta tarjoten luotettavuutta sekä tietoturvallisuutta IoT-sovelluksille. Se sopii erityisesti sellaisiin IoT-sovelluksiin, joissa IoT-laitteiden välinen viestintä sekä IoT-laitteiden ja palvelinten välinen viestintä tulee olla kaksisuuntaista sekä nopeaa. (Malik ja muut, 2018) Tämän lisäksi XMPP tukee eri IoT-sovelluksille tyypillisiä viestintämalleja, joihin kuuluu muun muassa asynkroninen viestintä sekä pyyntö/vastaus- ja julkaisija/tilaaja -mallit. XMPP-protokolla on skaalautuva ja tukee

laajalti IoT-laitteiden, -verkkojen ja -sovellusten yhteentoimivuutta. (XMPP, n.d.-b) Kuvassa 7 on esitetty XMPP-protokollalle tyypillinen viestintämalli, jossa viestit kulkevat palvelinten kautta asiakkaille. Yhdyskäytävän avulla XMPP-verkko voidaan yhdistää muihin verkkoihin. (Al-Fuqaha ja muut, 2015)



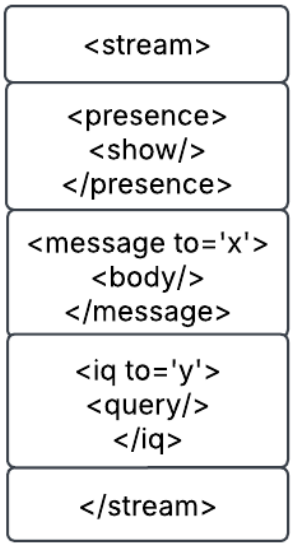
Kuva 7. Tyypillinen XMPP-viestintämalli, muokattu (Al-Fuqaha ja muut, 2015)

Avoimen lähdekoodin ansiosta XMPP-protokollaa voidaan laajentaa erilaisilla lisätoiminnallisuuksilla sovellusten asettamien vaatimusten täyttämiseksi. Laajennuksia voi tehdä joko itse tai käyttää XMPP:n XEP (XMPP Extension Protocols) -sarjassa määriteltyjä yleisesti käytettyjä laajennusprotokollia. (XMPP, n.d.-a)

XMPP käyttää hajautettua asiakas/palvelin -arkkitehtuuria reaaliaikaisen viestinnän mahdollistamiseksi asiakasohjelmien ja -laitteiden sekä palvelinten välillä. Jokaisella XMPP-verkon entiteetillä eli asiakkaalla tai palvelimella, on oma uniikki

sähköpostiosoitetta muistuttava JID (Jabber ID), jolla mahdollistetaan yksittäisten entiteettien tunnistaminen viestien välittämiseksi ja reitittämiseksi. (Saint-Andre, 2011)

XMPP-protokollan viestintä perustuu XML-säkeistöihin (engl. stanza), jotka ovat XML-pohjaisen viestinnän perusyksiköitä. XML-säkeistöt voidaan jakaa kolmeen eri tyyppiin, joita ovat `<message>`, `<presence>` ja `<iq>`. Entiteetit voivat lähettää viestejä toisilleen käyttämällä `<message>` -säkeistöä, jossa määritellään sekä lähettäjän että vastaanottajan JID ja viestin tyyppi. XML-säkeistöä `<presence>` käytetään puolestaan ilmaisemaan entiteetin käytettävyyttä verkossa. Entiteetin käytettävyys on hyödyllinen ominaisuus viestintää aloitettaessa, sillä kyseinen XML-säkeistö kertoo, voiko tietty entiteetti esimerkiksi vastaanottaa viestejä tällä hetkellä. `<iq>` -säkeistöä käytetään pyyntöjen tekemiseen esimerkiksi jonkin entiteetin hallitseman resurssin saamiseksi. (Saint-Andre, 2011) Kuvassa 8 on esimerkki XML-säkeistöstä koostuvasta XML-tietovirrasta.



Kuva 8. XML-säkeistöistä koostuva XML-tietovirta, muokattu (Al-Fuqaha ja muut, 2015)

Yhteyttä muodostettaessa käytetään TLS-salausprotokollaa viestinnän salaamiseksi, SASL (Simple Authentication and Security Layer) -viitekehystä asiakkaan todentamiseksi

sekä TCP-protokollaa kuljetuskerroksen protokollana. Asiakasohjelma tai -laite muodostaa yhteyden palvelimeen käyttäen TCP/IP-protokollaa ja avaa XML-tietovirran, joka mahdollistaa XML-elementtien, kuten XML-säkeistöjen lähettämisen muille entiteeteille. Asiakasohjelma tai -laite suorittaa TLS-kättelyn palvelimen kanssa viestinnän salaamiseksi, jonka jälkeen asiakas todennetaan SASL-viitekehyksen mekanismeilla, kuten `SCRAM-SHA-1`-mekanismilla. Asiakas kiinnittää resurssin muodostettuun istuntoon, mikä mahdollistaa useiden samanaikaisten istuntojen muodostamisen yhdelle entiteetille. Istunnon alettua asiakas ilmoittaa käytettävyytensä. Tämän jälkeen viestintä on mahdollista muiden entiteettien kanssa käyttäen eri XML-säkeistöjä. XMPP-protokolla mahdollistaa palvelinten välisen tiedonsiirron eri entiteettien viestien välittämiseksi. Palvelinten välisessä yhteydessä käytetään vastaavasti TLS- ja SASL-suojausta. XMPP-verkossa on erityisen tärkeää varmistaa tietoturvasyistä, että protokollat ovat seuraavassa järjestyksessä alhaalta ylös: TCP, TLS, SASL ja XMPP. (Saint-Andre, 2011)

TLS-suojauksen myötä tietojen luottamuksellisuus ja eheys toteutuu asiakkaan ja palvelimen tai palvelinten välisessä yhteydessä. Tämä auttaa suojautumaan muun muassa salakuuntelua ja väliintulohyökkäyksiä (engl. man-in-the-middle attack) vastaan. Johtuen XMPP-protokollan hajautetusta arkkitehtuurista viestintä koostuu pääosin useista eri XML-tietovirroista monien eri arkkitehtuurin entiteettien välillä. Tällöin kaikki XML-tietovirrat, joita käytetään viestin välittämiseen, eivät välttämättä ole TLS-suojattuja. Toinen tietoturvahaavoittuvuus liittyy TLS-kättelyyn. Ennen TLS-kättelyn onnistunutta toteutumista, viestin sisältämiä tietoja lähettäjästä ja vastaanottajasta voidaan mahdollisesti peukaloida hyökkääjän toimesta. (Saint-Andre, 2011)

Nastase on todennut Malikin ja muiden työssä (2018) TLS-suojauksen mahdollisen puuttumisen tietyistä XML-tietovirroista olevan merkittävä haavoittuvuus, joka voitaisiin ratkaista vahvalla päästä päähän -suojauksella. Sellaista ei ole kuitenkaan määritelty vielä Saint-Andren XMPP-standardissa (2011).

XMPP-protokollan laajennettavuuden myötä protokollalle on luotu päästä päähän -suojauksen mahdollistavia XEP-sarjassa määriteltyjä laajennusprotokollia, kuten esimerkiksi OMEMO (XEP-0384) (XMPP, n.d.-c). OMEMO-laajennusprotokolla varmistaa päästä päähän -suojauksen XMPP-verkossa ja turvaa tietojen eheyden, luottamuksellisuuden ja saatavuuden. OMEMO käyttää viestien salaamiseen ja salauksen purkamiseen Double Ratchet -algoritmia, jossa jokainen viesti suojataan satunnaisesti luodulla salausavaimella. Viestin salaamiseen käytetty salausavain salataan viestin vastaanottajan istuntoavaimella, ja välitetään viestin otsikkokentässä vastaanottajalle. Viestin vastaanottaja purkaa viestin salaamiseen käytetyn salausavaimen omalla istuntoavaimellaan, jonka jälkeen avainta voidaan käyttää viestin hyötykuorman eli viestin sisällön salauksen purkamiseen. (XMPP Standards Foundation, 2022)

4.4 AMQP

AMQP on avoimen standardin sovelluskerroksen protokolla, joka on suunniteltu luotettavaan binääristä viestintää hyödyntävään tiedonsiirtoon kahden osapuolen välillä (OASIS, 2012). AMQP-protokolla suunniteltiin alun perin väliohjelmaksi helpottamaan viestintää eri sovellusten ja järjestelmien välillä parantaen niiden yhteentoimivuutta. AMQP-protokollalla haluttiin lisätä viestinnän luotettavuutta ja skaalautuvuutta erityisesti rahoitusalalla. Viestintään tarvittiin avoin standardi, joka lisäisi luotettavuuden lisäksi myös viestinnän tehokkuutta. Ensimmäinen versio protokollasta julkaistiin vuonna 2006. (O'Hara, 2007) Viimeisin versio AMQP-protokollasta on vuonna 2012 julkaistu OASIS-standardoitu AMQP 1.0 (OASIS, 2012).

AMQP-protokolla soveltuu käytettäväksi IoT-sovelluksissa, sillä protokollaa voidaan käyttää useilla erilaisilla laitteilla samanaikaisesti AMQP-protokollan mahdollistaman yhteentoimivuuden ja skaalautuvuuden vuoksi. AMQP-protokolla tukee IoT-sovelluksille tyypillisiä viestintämalleja, kuten pisteestä pisteeseen -viestintää ja julkaisija/tilaaja -mallia. Esineiden internetissä tietoturva on oleellinen asia, joka toteutuu AMQP-

protokollassa viestinnän suojaamisella TLS-salausprotokollalla ja laitteiden todentamisella SASL-viitekehyksellä. (O'Hara, 2007; OASIS, 2012)

Sharma ja Gondhi ovat vertailleet työssään (2018) sovelluskerroksen protokollia ja todenneet AMQP-protokollan soveltuvan parhaiten liiketoiminta- ja kaupallisiin sovelluksiin. Työn perusteella AMQP-protokolla ei sovellu kaikkiin esineiden internetin käyttökohteisiin, sillä AMQP muun muassa kuluttaa huomattavasti enemmän virtaa kuin sovelluskerroksen protokolla MQTT. Tämän lisäksi AMQP:n viestintä ei ole reaaliaikaista kuten muiden sovelluskerrosten protokollien MQTT, CoAP ja XMPP viestintä.

AMQP-viestinnän osapuoliin kuuluu tuottaja (engl. producer), kuluttaja (engl. consumer) sekä välittäjä (engl. broker). Osapuolia kutsutaan säiliöiksi (engl. container), joista kukin edustaa yhtä AMQP-verkon päätepistettä. Tuottaja toimii viestien luojana sekä lähettäjänä, kun taas kuluttaja vastaanottaa ja prosessoi tuottajan lähettämiä viestejä. Välittäjän tehtävä on reitittää ja välittää viestit tuottajien ja kuluttajien välillä. Välittäjä pitää listaa jonoista, jotka ovat puskureita, joissa säilytetään yleensä yhdelle kuluttajalle välitettäviä viestejä tietyn aikaa. Välittäjän tehtäviin kuuluu pitää listaa myös aiheista, joita käytetään julkaisija/tilaaja -mallissa, jotta välittäjä osaa välittää kuluttajille niiden tilaamat aiheet. (OASIS, 2012)

AMQP käyttää linkkejä, joilla määritellään mihin suuntaan viestintä tapahtuu. Lähettäjälinkki muodostaa polun tuottajalta joko suoraan kuluttajalle tai vaihtoehtoisesti välittäjään, jonka kautta viestit reititetään ja välitetään oikealle kuluttajalle. Vastaanottajalinkki on välittäjän ja kuluttajan välinen linkki viestien vastaanottamiseen. Säiliöillä voi olla useita eri linkkejä. (OASIS, 2012)

AMQP-verkossa yhteyden muodostaminen tuottajan ja kuluttajan tai välittäjän välillä tapahtuu TCP-kuljetusprotokollalla. TCP-protokollan lisäksi yhteyden muodostaneet osapuolet voivat suorittaa TLS-kättelyn, jolloin TLS-salausprotokollaa käytetään TCP:n päällä yhteyden suojaamiseksi eli viestien luottamuksellisuuden ja eheyden

varmistamiseksi. SASL-viitekehyksen mekanismeja, kuten `ANONYMOUS` ja `PLAIN`, voidaan käyttää osapuolien todentamiseksi yhteyttä muodostettaessa. (OASIS, 2012) TLS-salausprotokollan ja SASL-viitekehyksien mekanismien käyttö ei ole kuitenkaan pakollista, mutta suositeltavaa. TLS ja SASL tulee konfiguroida oikein AMQP-verkossa tietoturvan lisäämiseksi.

Yhteyden muodostaneet osapuolet luovat istunnon yhteyden yli, jonka avulla varmistetaan viestien välittäminen oikeassa järjestyksessä. Istuntoja voi olla useampi osapuolten välillä. Osapuolet luovat linkit viestinnän suunnan määrittelemiseksi, ja linkit liitetään lähteeseen, josta viestit tulevat, ja kohteeseen, johon viestejä lähetetään. AMQP-verkko mahdollistaa kuluttajan ruuhkautumisen estämisen, jolloin kuluttaja voi määrittää, kuinka paljon viestejä se on valmis vastaanottamaan. AMQP tukee kolmea eri viestien toimitusmallia, joilla voidaan varmistaa, että viestit välitetään tuottajalta kuluttajalle halutulla tavalla. At-most-once -toimitusmallissa viesti välitetään kerran, eikä kuluttajalta vaadita kuittausta. At-least-once -toimitusmallissa viestiä välitetään kuluttajalle niin monta kertaa, kunnes kuluttaja kuittaa viestin vastaanottamisen. Exactly-once välittää viestin kuluttajalle vain kerran, jolloin vältytään saman viestin välittämiseltä useamman kerran. (OASIS, 2012)

McAteer ja muut ovat tutkineet työssään (2017) AMQP-protokollan tietoturvaa ja siihen vuosien aikana kohdistuneita tietoturvauhkia ja -haavoittuvuuksia. Heidän mukaansa merkittävän haavoittuvuuden AMQP-protokollaan luo protokollan avoin lähdekoodi, jonka ansiosta sovellukseen voidaan lisätä toiminnallisuuksia eri käyttökohteiden asettamien vaatimusten täyttämiseksi. Huonosti toteutetut lisätoiminnallisuudet AMQP-pohjaisen viestinnän järjestelmissä ja sovelluksissa saattavat luoda tietoturvariskin. Työssä on todettu TLS-suojauksen olevan purettavissa AMQP-verkossa, jolloin hyökkääjät voivat siepata ja purkaa AMQP-verkon osapuolten välistä viestintää, mikäli heillä on tarpeeksi aikaa ja resursseja, kuten esimerkiksi laskentatehoa. Tästä voidaan päätellä, että TLS-suojaus ei ole täysin läpäisemätön, vaikka se tarjoaakin lähtökohtaisesti vahvan suojauksen viestinnälle.

McAteerin ja muiden työssä (2017) on esitelty myös vuonna 2017 tapahtunut palvelunestohyökkäys, jossa yli 5000 laitetta saatiin siepattua murtamalla yhden heikolla salasanalla varustetun IoT-laitteen suojauksen. Yhden murretun IoT-laitteen kautta voitiin lähettää IoT-laitteista koostuvaan verkkoon haittaohjelma, joka levisi laitteelta toiselle. Sama uhka voi kohdistua myös AMQP-verkkoihin, mikäli AMQP-palvelinten väliset yhteydet saadaan vaarannettua. Näin ollen on tärkeää konfiguroida AMQP-protokollan sekä AMQP-protokollaa käyttävien laitteiden asetukset vastaamaan riittäviä tietoturvastandardeja.

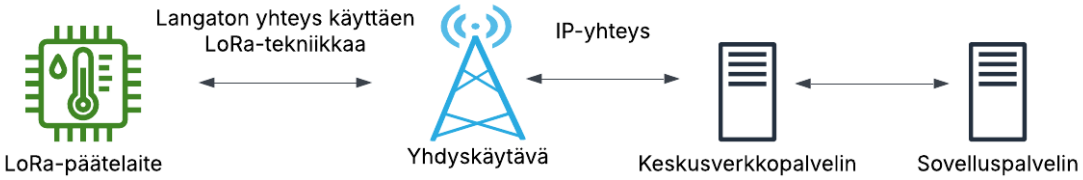
4.5 LoRaWAN

LoRaWAN (Long Range Wide Area Network) on laajan alueen vähätehoinen tiedonsiirtoverkko, jonka tarkoituksena on yhdistää etenkin paristokäyttöisiä IoT-laitteita tai muita IoT-esineitä internetiin langattomasti. LoRaWANissa on otettu huomioon IoT:n vaatimukset, joihin kuuluu muun muassa vähäinen virrankulutus, kaksisuuntainen viestintä, päästä päähän -tietoturvallisuus sekä useamman eri laitteen liittäminen samaan verkkoon. LoRaWAN määrittää IoT-protokollapinon siirtokerroksen MAC-kerroksen ja toimii LoRa-tekniikan määrittämän fyysisen kerroksen päällä. LoRaWAN tunnetaan sen pitkästä kantamasta, joka on jopa 15 kilometriä maaseutualueilla. LoRaWANin käyttää toimiakseen lisensoimattomia radiotaajuuksia. (LoRa Alliance, n.d.)

LoRa on lyhenne sanoista long range eli pitkä kantama. LoRa on Semtechin kehittämä chirp-hajaspektriteknologiaan (engl. Chirp Spread Spectrum) perustuva hajaspektrimodulaatiotekniikka (engl. Spread Spectrum Modulation). LoRa-tekniikkaa käytetään muodostamaan vähän virtaa kuluttavan pitkän kantaman langaton yhteys radiotaajuuksilla LoRaWAN-ekosysteemin fyysisessä kerroksessa. (Semtech, n.d.)

LoRaWANin infrastruktuuri muodostuu päätelaitteista, yhdyskäytävistä ja keskusverkkopalvelimesta. LoRaWAN-tiedonsiirtoverkko käyttää tähtiverkkoon perustuvaa star-of-stars-verkkotopologiaa. Verkkotopologiassa yhdyskäytävä toimii viestien välittäjänä päätelaitteiden ja keskusverkkopalvelimen välillä. Yhdyskäytävät

yhdistetään keskusverkkopalvelimeen tavallisia IP-yhteyksiä käyttäen. Yhdyskäytävät muuntavat radioaaltojen sisältämät viestit IP-paketeiksi ja toisin päin. Päätelaitteet eivät näin ollen siis itse käytä IP-yhteyksiä, vaan yhteys toteutetaan radiotaajuuksilla päätelaitteilta yhdyskäytäville. LoRaWANin tuki kaksisuuntaiselle viestinnälle ja ryhmälähetykselle (engl. multicast) tehostaa radiotaajuusalueen käyttöä muun muassa langattomia ohjelmistopäivityksiä tehdessä. (LoRa Alliance, n.d.) Kuvassa 9 on esitetty LoRaWANin infrastruktuuri.



Kuva 9. LoRaWANin infrastruktuuri, muokattu (LoRa Alliance, n.d.)

LoRaWANin kestävä ja skaalautuva infrastruktuuri tekee toiminnasta ja datan keruusta tehokkaampaa. LoRaWANin käyttökohteita on lukuisia johtuen sen mahdollistamasta kehittyneestä resurssienhallinnasta, etävalvonnasta sekä automaatiomahdollisuuksista eri aloilla. LoRaWANia käytetään erityisesti maataloudessa, rakennuksissa ja kaupungeissa, teollisuudessa sekä logistiikassa. LoRaWAN erottuu muista tiedonsiirtoverkoista etenkin alhaisilla kustannuksilla, skaalautuvuudella ja käyttöönoton joustavuudessa, joten sitä suositaan monissa IoT:n käyttökohteissa. Esimerkiksi NB-IoT tarjoaa LoRaWANia korkeammat tiedonsiirtonopeudet ja pienemmän viiveen, mutta edellä mainitut LoRaWANin hyödylliset ominaisuudet ovat tärkeämpiä useassa IoT:n eri käyttökohteessa. (LoRa Alliance, n.d.)

Gemalton ja muiden (2017) mukaan LoRaWAN-protokollan tietoturvaominaisuudet on suunniteltu vastaamaan LoRaWANille asetettuja yleisiä kriteerejä, joita ovat vähäinen virrankulutus, toteutuksen ja käyttöönoton helppous, alhaiset kustannukset ja skaalautuvuus. LoRaWAN tarjoaa käyttäjille yhteyden päästä päähän -suojauksen, jota

kaikissa protokollissa ei ole. LoRaWAN takaa viestien hyötykuorman olevan salattu aina päätelaitteelta palvelimelle asti. LoRaWAN käyttää tietoturvaominaisuuksien toteuttamiseen standardisoituja sekä testattuja algoritmeja, jotka soveltuvat käytettäviksi rajoitetuissa laitteissa ja verkoissa.

Jokaisella LoRaWAN-laitteella on uniikki 128-bittinen AES (engl. Advanced Encryption Standard) -avain, jota kutsutaan nimellä AppKey. Tämän lisäksi laitteilla on globaali tunniste. Laitteelle myönnettyä avainta ja tunnistetta käytetään laitteen todentamisprosessissa. Yhteyden muodostaminen päätelaitteen liittyessä verkkoon tietoturvallisesti tapahtuu siten, että sekä päätelaitteella, että verkolla on tieto päätelaitteen avaimesta. Tämän jälkeen muodostetaan kaksi eri istuntoavainta: NwkSKey ja AppSKey. Istuntoavainten muodostaminen mahdollistaa tietoturvan jakamisen kahdelle tasolle, jotta tiedot pysyvät luottamuksellisina, vaikka LoRaWAN-verkko vaarantuisi. (Gemalto ja muut, 2017)

NwkSKey-avaimen tehtävä on todentaa päätelaitteen ja verkon väliset viestit, jotta viesteihin ei pääse käsiksi sellaiset osapuolet, joilla ei ole valtuuksia viesteihin. NwkSKey-avaimella varmistetaan viestien aitous ja eheys. AppSKey-avaimella toteutetaan päästä päähän -suojaus viestin hyötykuorman tiedoille. AppSKey-avain estää muita osapuolia, kuten operaattoreita tai muita välittäjiä, lukemasta viestin hyötykuormaa eli sen sisältöä, koska avain voidaaan piilottaa muilta osapuolilta. Avainta käytetään tietojen salaamiseen ja tietojen purkamiseen päätelaitteen ja verkon välisessä viestinnässä. NwkSKey- ja AppSKey -avainten lisäksi LoRaWAN käyttää MIC (Message Intergrity Code) -koodia. Jokainen viesti sisältää MIC (Message Intergrity Code) -koodin, jolla varmistetaan, ettei viestiä ole peukaloitu kenenkään toimesta. Kahden tietoturvakerroksen ja MIC-koodin myötä LoRaWAN ei tarvitse erillistä salausprotokollaa, kuten TLS-protokollaa, mikä lisäisi virrankulutusta, monimutkaisuutta ja toteutukseen liittyviä kuluja. (Gemalto ja muut, 2017)

Coman ja muut (2019) ovat tutkineet työssään LPWAN (Low Power Wide Area Network) -teknologioiden haavoittuvuuksia. Yksi yleisimmistä tietoturvauhista LPWAN-verkoissa on työn perusteella hyökkääjän fyysinen pääsy verkon osapuoliin, kuten laitteisiin, palvelimiin tai yhdyskäytäviin. Tällöin hyökkäyksen kohteeksi joutunut laite, muu osapuoli tai koko verkko on vaarantunut. Gemalton ja muiden (2017) mukaan laitteen uniikkia AppKey- ja yhteyden muodostuessa syntyviä istuntoavaimia säilytetään lähtökohtaisesti LoRaWAN-päätelaitteessa. LoRaWANia käytettäessä suositellaankin AppKey- ja istuntoavainten säilyttämistä peukaloinnin kestävässä laitteistokomponentissa (Secure Element), joka estää hyökkääjän fyysisen pääsyn avaimiin (Gemalto ja muut, 2017). Comanin ja muiden (2017) mukaan LoRaWAN on altis myös häirinnälle (engl. jamming) sen käyttäessä pieniä taajuuskaistoja.

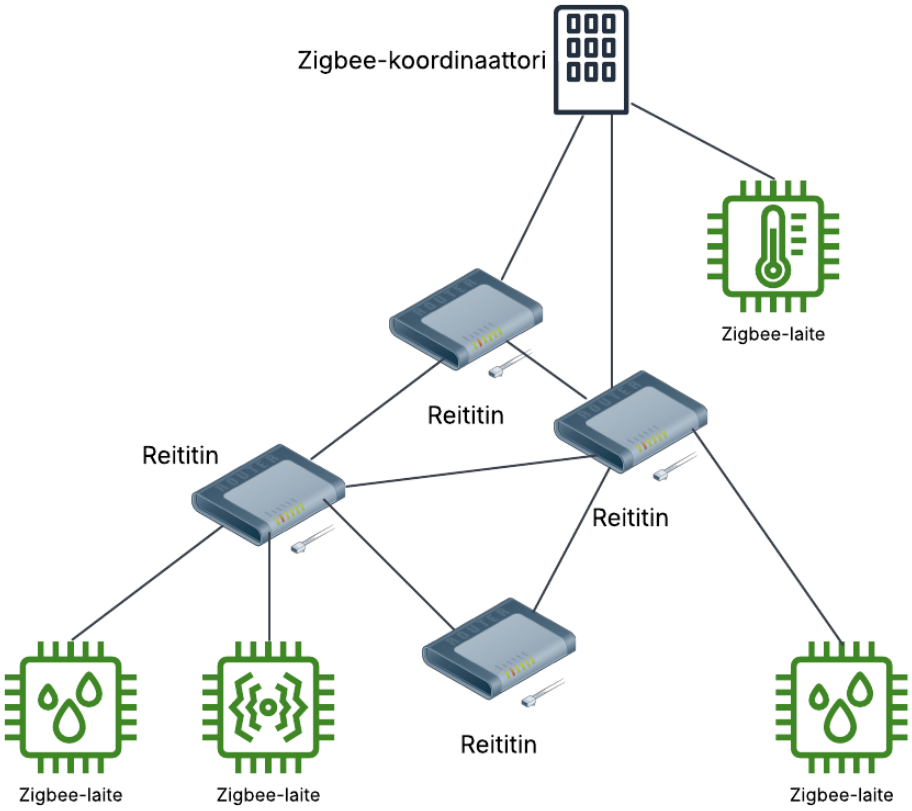
Coman ja muut (2019) esittävät työssään myös MIC-koodin väsytyshyökkäyksen (engl. brute force attack), jossa MIC-koodi voitaisiin teoriassa murtaa järjestelmällisesti, jolloin viestintää voisi peukaloida. Käytännössä tämä on kuitenkin hyvin harvinaista, koska MIC-koodi on neljä tavua pitkä, ja hyökkääjän tulisi lähettää yli 4 miljardia pakettia murtaakseen MIC-koodin. Toisaalta hyökkäystä voidaan nopeuttaa lähettämällä paketteja rinnakkain useilla kanavilla ja useita yhdyskäytäviä käyttämällä.

4.6 Zigbee

Zigbee on langattomaan kaksisuuntaiseen viestintään suunniteltu protokolla, joka tunnetaan sen alhaisista kustannuksista sekä virrankulutuksesta. Zigbee määrittää verkko- ja sovelluskerroksen IoT-protokollapinon fyysisen ja siirtokerroksen MAC-alikerroksen päälle. Fyysinen ja MAC-kerros käyttävät IEEE 802.15.4 standardin mukaisia radioyhteyksiä ja taajuusalueita. Fyysinen kerros käyttää taajuusalueita 868-915 MHz sekä 2.4 GHz, joista 868 MHz toimii Euroopassa, 915 MHz -taajuus on käytössä Australiassa ja Yhdysvalloissa, sekä korkein taajuusalue 2.4 GHz on käytössä maailmanlaajuisesti. (Zigbee Alliance, 2015)

Zigbee soveltuu käytettäväksi IoT-sovelluksiin, joissa vaaditaan luotettavaa ja alhaisen virrankulutuksen tiedonsiirtoa ilman suuria tiedonsiirtonopeuksia. Zigbee-protokollaa käytetään etenkin koti- ja rakennusautomaatiossa. Tämän lisäksi Zigbee-protokollaa käytetään terveydenhuollossa ja teollisuusautomaatiossa etäseurantalaitteissa sekä erilaisissa mittareissa. Zigbeen tukeman mesh-verkkotopologian myötä laitteet voivat kommunikoida helposti keskenään parantaen verkon kattavuutta ja luotettavuutta. Mesh-verkko on hyödyllinen monessa eri IoT-sovelluksessa muun muassa skaalautuvuutensa ansiosta. (Zigbee Alliance, 2015; Khanji ja muut, 2019)

Zigbeen verkkokerroksessa käyttämä verkkotopologia voi olla joko tähti-, puu- tai mesh-verkkotopologia. Verkkoa hallitsee tähtitopologiaa käytettäessä yksi laite, jota kutsutaan Zigbee-koordinaattoriksi, jonka vastuulla on hallita verkon laitteita. Tähtitopologia on yksinkertainen, mutta rajoittunut kantamaltaan. Puuverkkotopologiassa Zigbee-koordinaattori määrittää verkon parametrit, kuten taajuusalueen, ja käynnistää verkon. Puuverkkotopologiassa käytetään reitittimiä tiedon välittämiseen muodostaen hierarkkisen reitityksen. Kyseisen topologian avulla saadaan lisättyä Zigbee-verkon kantamaa, mutta sen käyttäminen saattaa lisätä myös viivettä. Mesh-verkkotopologiassa laitteet pystyvät kommunikoimaan keskenään muodostaen useita eri polkuja tiedonsiirrolle parantaen Zigbee-verkon tiedonsiirron luotettavuutta ja skaalautuvuutta. (Zigbee Alliance, 2015) Kuvassa 10 on esitetty Zigbeen verkkokerroksen mesh-verkkotopologia.



Kuva 10. Zigbeen verkkokerroksen mesh-verkkotopologia, muokattu (Khanji ja muut, 2019)

Zigbeen sovelluskerroksen viitekehys toimii Zigbeen verkkokerroksen päällä ja koostuu application support sub-layer (APS) -kerroksesta sekä Zigbee-laiteobjekteista. APS-kerros toimii rajapintana verkkokerroksen ja sovelluskerroksen välillä tarjoten palveluita tiedonsiirtoon, tietoturvaan ja laitteidenhallintaan. APS-kerros hoitaa laitteiden välisen tiedonsiirron ottaen huomioon pakettien uudelleenjärjestämisen ja tiedon jakamisen pienempiin osiin. APS-kerros muodostaa loogisia linkkejä laitteiden välille niiden toiminnallisuuksien perusteella, ja mahdollistaa laitteiden löytämisen verkossa. Zigbee-laiteobjektin tehtäviin kuuluu laitteen roolien hallinta, verkkokonfiguraatio ja tietoturva. Zigbee-laiteobjektia käytetään muun muassa verkkoon liityttäessä ja verkkoa muodostettaessa. Tämän lisäksi laiteobjektilla voidaan tehdä kyselyjä muiden laitteiden resurssien käyttämiseksi. Zigbee-sovelluskerroksessa voidaan määritellä laitteille sovellusprofiileja, joissa on määritelty toiminnallisuudet jotakin tiettyä käyttökohdetta,

kuten esimerkiksi kotiautomaatiota, varten. Sovellusprofiili koostuu klustereista, joissa on attribuutteja ja komentoja jonkin toiminnon suorittamiseksi, kuten esimerkiksi lämpötilan havaitseminen. (Zigbee Alliance, 2015)

Zigbee käyttää AES-128 (Advanced Encryption Standard) -standardia tietojen salaamiseen varmistaen tietojen luottamuksellisuuden sekä eheyden. Jokainen Zigbee-viesti sisältää MIC-koodin, jolla varmistetaan, ettei viestiä ole peukaloitu viestin välittämisen aikana. Zigbee-verkossa on turvakeskus (engl. trust center), joka on yleensä Zigbee-verkon Zigbee-koordinaattori. Turvakeskuksen tehtävänä on varmistaa laitteiden välisten yhteyksien päästä päähän- suojaus sekä hallita verkon avaimia, joita käytetään Zigbee-verkon tietoturvan muodostamiseen. Zigbee-verkkoon liittyvät laitteet todennetaan turvakeskuksen avulla. (Khanji ja muut, 2019)

Zigbee käyttää kolmea erilaista avainta, joita ovat master-, verkko- ja linkkiavain. Master-avain asennetaan joko valmistusvaiheessa tai vaihtoehtoisesti turvakeskuksen toimesta. Master-avainta ei kuitenkaan varsinaisesti käytetä tietojen salaamiseen, vaan sitä käytetään linkkiavainten muodostamiseen. Linkkiavain on kahden laitteen käyttämä avain sovelluskerroksessa tapahtuvan viestinnän suojaamiseksi. Verkkoavain usein jaetaan kaikille laitteille Zigbee-verkossa, ja sillä suojataan verkkokerroksessa tapahtuva viestintä. Zigbee-verkon käyttämät avaimet takaavat tietojen aitouden varmistamisen ja eheyden niin tiedon lähteessä, kuin tiedon määränpäässä. (Khanji ja muut, 2019)

Khanji ja muut ovat tutkineet työssään (2019) Zigbee-protokollan tietoturvaongelmia. Heidän mukaansa Zigbee-verkossa lähetettyjen tietojen luottamuksellisuus ja niiden aitouden varmistaminen vaarantuu, mikäli hyökkääjät pääsevät käsiksi Zigbee-verkon osapuolten tunnistetietoihin käsiksi. Xu ja muut ovat esittäneet Khanjin ja muiden työssä (2019) tavan välttyä ongelmalta salausavainten päivittämisellä aikasynkronointia hyödyntäen. Toinen työssä esitelty hyökkäys keskittyy väliintulohyökkäykseen, jossa hyökkääjä häiritsee verkon muodostamista tai vaihtoehtoisesti levittää vääriä reititystietoja. Ongelman ratkaisemiseksi Shang ja muut ovat ehdottaneet Khanjin ja

muiden työssä (2019) uutta tapaa jakaa salausavaimet verkossa Diffie-Hellmanin algoritmilla ja käyttämällä kättelyprotokollaa uusien laitteiden liittyessä Zigbee-verkkoon.

Khanji ja muut (2019) käsittelevät työssä myös Zigbee-verkossa lähetettäviä väärennettyjä paketteja, joiden avulla voidaan toteuttaa palvelunestohyökkäys tai vastaavasti johtaa tiedon vastaanottaja harhaan. Tämän lisäksi työssä kiinnitetään huomiota Zigbeen sovellusprofiileihin, joihin voidaan kohdistaa hyökkäyksiä vahingoittaen sovellusprofiilien klustereita. Edellä lueteltujen haavoittuvuuksien perusteella voidaan todeta Zigbeen olevan altis erilaisille hyökkäyksille, vaikka Zigbeen tietoturva onkin yleisesti ottaen oletusasetuksilla tarpeeksi vahva. Tietoturvaa voidaan näin ollen joutua lisäämään eri tavoilla, kuten uudella tavalla jakaa salausavaimet, mikäli verkossa käsitellään esimerkiksi arkaluontoisia tietoja.

4.7 NB-IoT

NB-IoT on lyhenne termistä NarrowBand-Internet of Things. Se on jo olemassa oleviin standardeihin perustuva esineiden internetiä varten suunniteltu LPWA (engl. low power wide area) -teknologia, jonka tarkoituksena on mahdollistaa IoT-laitteiden käyttäminen ja IoT-palveluiden luominen. NB-IoT erottuu muista teknologioista etenkin sen käyttämien IoT-laitteiden akunkeston myötä, joka voi olla jopa yli 10 vuotta eri käyttötarkoituksissa. NB-IoT tuo parannuksia laitteiden virrankulutuksen lisäksi järjestelmän kapasiteetin kasvattamiseksi ja radiotaajuusalueen tehokkaaksi käyttämiseksi. (GSM Association, n.d.) NB-IoT-teknologia on spesifioitu 3GPP:n toimesta vastaamaan kasvavaan kysyntään LPWA-teknologialle (GSM Association, 2019).

NB-IoT on suunniteltu toimimaan niin maaseudulla, kuin sisätiloissa, joissa kuuluvuus voi olla heikko. NB-IoT tuo yksinkertaisuutta IoT-laitteisiin, ja NB-IoT-teknologiaa tukee valtaosa mobiililaite-, piirisarja- moduulivalmistajista. NB-IoT-teknologiaa voidaan käyttää rinnakkain jo olemassa olevien mobiiliverkkojen, kuten 4G-verkon kanssa, ilman uuden infrastruktuurin muodostamista. Samalla NB-IoT hyötyy mobiiliverkkoihin

rakennetuista tietoturvaominaisuuksista, jotka muun muassa mahdollistavat tietojen luottamuksellisuuden ja eheyden, sekä laitteiden todennuksen. (GSM Association, n.d.)

NB-IoT tukee suurta määrää pienen tiedonsiirtonopeuden laitteita samanaikaisesti sekä NB-IoT-teknologiaa käyttävät laitteet ovat kustannuksiltaan edullisia. NB-IoT-laitteet toimivat optimoidussa verkkoarkkitehtuurissa, joka on käytettävissä maailmanlaajuisesti, ilman uusien infrastruktuurien rakentamista. (GSM Association, 2019) Näin ollen NB-IoT on skaalautuva ja tietoturvallinen tehden siitä sopivan erilaisiin IoT:n käyttökohteisiin, kuten esimerkiksi älykkääseen maanviljelyyn, mittaamiseen tai teolliseen esineiden internetiin.

Kuten NB-IoT:n (NarrowBand-IoT) nimestä voi päätellä, se käyttää kapeaa taajuuskaistaa, joka on 180 kHz sekä lähetys-, että vastaanottosuunnassa. Kapea taajuuskaista mahdollistaa NB-IoT:n toimimisen rajoitetuilla taajuusalueilla. Se sopii hyvin pienten tietomäärien lähettämiseen, kuten esimerkiksi anturien keräämien tietojen lähettämiseen. NB-IoT voidaan ottaa käyttöön määräämällä sille oma taajuusalue, jolloin taajuuskaistaa ei jaeta muiden verkkojen kanssa toimien itsenäisesti. Tätä kutsutaan standalone-käyttöönotoksi. In-band-käyttöönotossa NB-IoT jakaa jonkin LTE-verkon taajuuskaistoista, ja käyttää sitä toimiakseen. Guardband-käyttöönotossa NB-IoT käyttää LTE-verkon LTE-kantoaaltojen välissä olevia vapaita taajuusalueita toimiakseen. (GSM Association, 2019) Näin ollen NB-IoT voidaan ottaa käyttöön usealla eri tavalla riippuen käyttökohteen asettamista vaatimuksista ja alueella saatavilla olevasta mobiiliverkkoinfrastruktuurista.

NB-IoT-laitteiden alhainen virrankulutus ja pitkä akunkesto toteutetaan NB-IoT:n PSM (Power Saving Mode) - ja eDRX (Extended Discontinuous Reception) -ominaisuuksilla. PSM-ominaisuuden avulla NB-IoT-laitteet voivat siirtyä syvään lepotilaan, kun ne eivät lähetä tai vastaanota tietoja ilman, että laitteen tarvitsisi poistua verkosta. PSM-tilaan siirtyminen tapahtuu sen jälkeen, kun laite on vastaanottanut tai lähettänyt tietoja, ja sopinut verkon kanssa, kuinka pitkään laite voi olla syvässä lepotilassa. Tämä vähentää

energiankulutusta etenkin laitteissa, jotka lähettävät harvoin dataa. eDRX-ominaisuus on toinen virransäästöä parantava ominaisuus, joka mahdollistaa sen, että laite voi olla kuuntelematta verkkoa tietyn ajan. Tällöin laite sammuttaa radionsa hetkellisesti, IoT-sovelluksissa usein muutamiksi sekunneiksi, ja kuluttaa näin ollen vähemmän virtaa. (GSM Association, 2019)

NB-IoT:n tietoturva riippuu pitkälti käytössä olevien mobiiliverkkojen tietoturvaominaisuuksista. Usein mobiiliverkkojen tietoturvaominaisuudet mahdollistavat käyttäjien tietojen suojaamisen, laitteiden todentamisen ja tunnistamisen, sekä tietojen luottamuksellisuuden ja eheyden varmistamisen (GSM Association, n.d.). GSM Associationin oppaassa NB-IoT:n käyttöönotosta (2019) mainitaan, että NB-IoT:n päästä päähän -suojaus ei välttämättä toteudu, mikäli kaikkia vaihtoehtoisia NB-IoT:n ominaisuuksia ei ole otettu käyttöön. Oppaassa on esitelty BEST (Battery Efficiency Security for Low Throughput) -ominaisuus, jonka avulla viestin hyötykuorma voidaan salata ilman liiallista viestinnällistä kuormitusta, mikä on tärkeä asia IoT-laitteissa. BEST-ominaisuus käyttää symmetrisiä salausavaimia, jotka voidaan välittää eteenpäin esimerkiksi TLS-suojausta varten, tai niitä voidaan käyttää tietojen eheyden ja viestinnän varmistamiseksi. Näin ollen tämänkin teknologian kohdalla on tärkeää muistaa konfiguroida verkko ja laitteet oikein, tai lisätä TLS-suojaus, tai muu salausprotokolla tietoturvan lisäämiseksi.

Coman ja muut ovat tutkineet työssään (2019) myös NB-IoT:n tietoturvaa. LoRaWANin tapaan myös NB-IoT on haavoittuvainen häirinnälle, sillä se käyttää pientä 180 kHz taajuuskaistaa. Työssä on esitetty myös NB-IoT-verkkoon kohdistuva hyökkäys, jossa Android-pohjaista NB-IoT-SIM-kortilla varustettua päätelaitetta käytetään lähettämään haitallisia tietoja muille verkon osapuolille. Yksityiseen NB-IoT-verkkoon yhdistetyn päätelaitteen avulla samaan verkkoon voitiin liittää tietokone verkon skannaamiseksi ja muiden verkossa olevien laitteiden IP-osoitteiden paljastamiseksi. Saatuaan IP-osoitteet hyökkääjä olisi voinut lähettää haitallista dataa verkosta löytämiinsä laitteisiin, tai pakottaa laitteet vastaanottamaan ja lähettämään dataa jatkuvasti

palvelunestohyökkäyksen toteuttamiseksi. Näin ollen NB-IoT ei ole täysin tietoturvallinen, vaan siihen voidaan kohdistaa erilaisia hyökkäyksiä, ja sen tietoturvaan voidaan vaikuttaa vaihtoehtoisilla NB-IoT-ominaisuuksilla.

4.8 Thread

Thread on Thread Groupin vuonna 2015 julkaisema IoT-laitteiden langattomaan viestintään suunniteltu matalan tehon ja alhaisten kustannusten protokolla. Thread perustuu olemassa oleviin avoimiin standardeihin, kuten IEEE 802.15.4- ja IPv6-standardiin. Protokolla on tarkoitettu lähinnä IoT-sovelluksiin kotiautomaatiossa sekä kaupallisissa sovelluksissa, joissa halutaan käyttää IP-protokollaa ja jotain lukuisista sovelluskerroksen protokollista. (Unwala ja muut, 2019; Thread Group, 2022)

Thread-verkon muodostaminen ja ylläpitäminen on vaivatonta sen mesh-verkkotopologian ansiosta, jossa laitteet voivat välittää tietoa toisilleen ja konfiguroida itse itsensä. Mesh-verkkotopologia mahdollistaa myös sen, että viestintä laitteiden välillä on aina luotettavaa ja tietoturvallista, vaikka yksittäiset laitteet lakkaavat toimimasta. Thread-verkon viestintä on aina salattua ja turvallista, eikä laitteen anneta liittyä verkkoon ilman riittäviä valtuuksia. Thread on skaalautuva ja tukee käyttökohteen mukaan muutamia laitteita aina tuhansiin laitteisiin asti. Thread on pääasiassa verkkokerroksen protokolla, jonka päällä voidaan käyttää useita eri sovelluskerroksen protokollia. (Thread Group, 2022)

Thread on rakennettu IEEE 802.15.4 standardin mukaisten fyysisen kerroksen ja siirtokerroksen MAC-kerroksen päälle mahdollistaen 250 kbps -tiedonsiirtonopeuden 2.4 GHz -taajuuskaistaa käyttäen. Standardin mukainen fyysinen kerros ja MAC-kerros mahdollistavat luotettavan viestinnän laitteiden välillä. Thread käyttää IPv6-protokollaa osoitteiden muodostamiseen laitteille, viestien reitittämiseen, sekä integroimiseen olemassa olevien IP-pohjaisten verkkojen sekä internetin kanssa. 6LoWPAN-teknologiaa käytetään IPv6-pakettien tehokkaaseen siirtämiseen fyysisen ja MAC-kerroksen päällä. 6LoWPAN mahdollistaa matalan tehon laitteiden ja pienen kaistanleveyden verkkojen

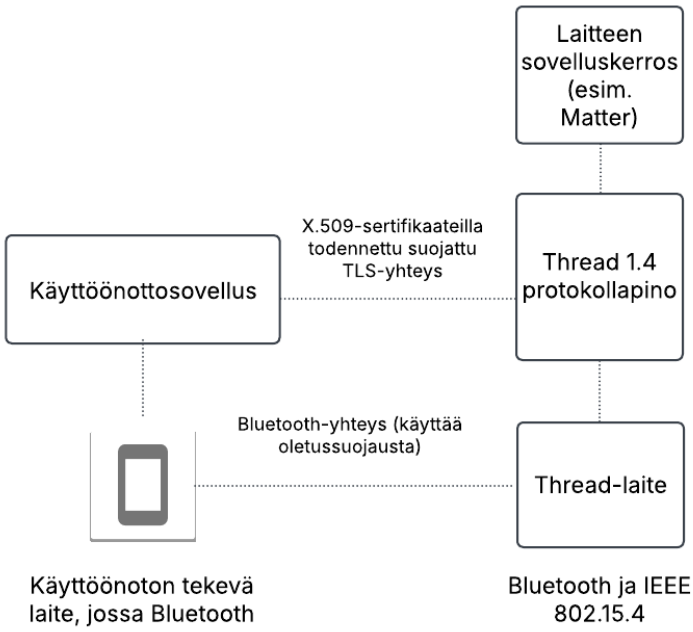
käyttämisen. Thread käyttää pääsääntöisesti UDP-kuljetusprotokollaa IPv6-protokollan päällä verkkoa muodostettaessa ja ylläpidettäessä, mutta mahdollistaa myös TCP-protokollan käytön sovelluskerroksen viestinnässä. (Thread Group, 2022)

Thread-verkossa käytetään reitittimiä Thread-laitteiden viestinnän reitittämiseen. Niiden tehtäviin kuuluu myös uusien laitteiden liittäminen Thread-verkkoon tietoturvallisesti. Reitittimien lisäksi Thread-verkossa on REED (Router Eligible End Device) -laitteita, jotka voivat tarpeen mukaan toimia reitittiminä, sekä FED (Full End Device) -laitteita, jotka ovat samankaltaisia kuin REED-laitteet, mutta eivät voi toimia reitittiminä. Yksi reitittimistä valitaan johtajaksi (engl. leader), jonka vastuulla on tehdä verkkoon liittyviä päätöksiä esimerkiksi siitä, mikä REED-laitteista valitaan reitittimeksi. Thread-verkon reitittimien määrä määrittää Thread-verkon topologian. Jos käytössä on yksi reititin, käytetään tavallista tähtitopologiaa. Useampi reititin Thread-verkossa muodostaa automaattisesti mesh-verkon. (Thread, 2022)

Thread-verkkoon kuuluu myös erilaisia minimalistisia laitteita (Minimal Thread Devices), jotka voivat kommunikoida ainoastaan niiden isäntäreitittimen kanssa. Thread-verkossa on reunareitittimiä, joilla Thread-verkko liitetään rinnakkaisiin verkkoihin, jotka käyttävät fyysisessä kerroksessa eri standardin mukaista teknologiaa. Tällainen verkko voisi olla esimerkiksi WiFi-verkko. (Thread Group, 2022)

Kun uutta Thread-laitetta liitetään Thread-verkkoon, laitteen täytyy ensin etsiä reititin, joka voi liittää laitteita verkkoon (Joiner Router), ja muodostaa yhteys reitittimen kanssa. Tämän jälkeen laitteen käyttöönotto eli Commissioning voidaan aloittaa. Käyttöönottovaiheessa verkkoon liittyvä laite todennetaan, ja sille annetaan verkon tunnistetiedot, jotka laite säilyttää. Saatuaan verkon tunnistetiedot, laite lähettää pyyntöjä usealle eri Thread-verkon reitittimelle. Jokin reitittimistä vastaa laitteen pyyntöön ja liittää laitteen verkkoon muodostaen turvallisen yhteyden sekä määrittäen IPv6-osoitteet. Laite säilyttää verkon tunnistetiedot, jotta käyttöönottovaihetta (Commissioning) ei tarvitse tehdä uudelleen. (Thread Group, 2022)

Thread 1.4-versiossa käyttöönottovaiheen tietoturvallisuutta on parannettu TLS-suojauksen avulla useiden Thread-laitteiden liittämiseen verkkoon nopeasti ja tietoturvallisesti. Vahva suojaus pohjautuu todennettuun, varmennepohjaiseen TLS-istuntoon Thread-laitteen ja käyttöönottovaiheen tehneen laitteen kanssa. Ominaisuutta kutsutaan nimellä Thread Commissioning over Authenticated TLS (TCAT). Ominaisuus mahdollistaa muun muassa verkon tunnistetietojen tietoturvallisen välittämisen verkossa. (Thread Group, 2024) Kuvassa 11 on mallinnettu TCAT-ominaisuutta uuden Thread-verkkoon liitettävän laitteen käyttöönotossa, jossa jo käyttöönoton tehnyt laite toimii käyttöönoton tekevänä laitteena verkkoon liittyvälle laitteelle.



Kuva 11. TCAT-käyttöönotto, muokattu (Thread Group, 2024)

Thread Groupin mukaan (2022) Thread-protokolla perustuu olemassa oleviin avoimiin standardeihin. Tästä voidaan päätellä, että protokollan tietoturva riippuu jossain määrin protokollan käyttämien standardien tietoturvasta. Tietoturvaominaisuuksia on kuitenkin toteutettu Thread-protokollassa itsessään muun muassa käyttöönottovaiheessa, jota on

parannettu uusimmassa Thread 1.4-versiossa TLS-suojauksella. Thread Group on julkaissut uusimman version protokollastaan vuonna 2024, joten sen tietoturvan voidaan päätellä olevan nykyisten IoT-laitteiden ja -järjestelmien asettamien tietoturvavaatimusten mukainen.

5 Johtopäätökset ja yhteenveto

Tutkielman tarkoituksena oli esitellä kirjallisuuskatsauksen avulla IoT-laitteissa yleisimmin käytetyt protokollat sekä niiden toiminta, ja selvittää, mihin käyttötarkoitukseen ne ovat, ja miksi niitä tarvitaan IoT-infrastruktuurissa.

Valtaosa IoT-protokollista on suunniteltu nopeaan, tehokkaaseen ja jopa reaaliaikaiseen tiedonsiirtoon. Protokollat tulee valita käyttökohteen vaatimusten mukaan, sillä protokollia on erilaisia, ja kukin protokolla priorisoi eri IoT-ympäristön vaatimuksia. Protokollan valinta riippuu myös siitä, halutaanko IoT-sovelluksessa käyttää protokollaa pelkästään esimerkiksi nopeaan viestintään, vai vaatiiko käyttötarkoitus myös oman verkkonsa, joka olisi tietoturvallisempi ratkaisu.

IoT-protokollat ovat pääasiassa tietoturvaominaisuuksiltaan puutteellisia, ja ne vaativat muita protokollia tai teknologioita lisäksi toimiakseen tietoturvallisesti. Tietoturvaa lisäävät protokollat vaikuttavat IoT-laitteiden suorituskykyyn protokollien vaatiessa enemmän laskentatehoa laitteelta. Tietoturvaa lisäävät protokollat lisäävät myös viestinnällistä kuormitusta johtuen salausprotokollien käyttämistä menetelmistä salatun yhteyden muodostamiseksi, jolloin kaistanleveyttä käytetään enemmän.

Tietoturvan parantamiseksi IoT-sovelluksissa ja laitteissa tulisi käyttää suojattuja yhteyksiä, jotka voidaan muodostaa käyttämällä salausprotokollia kuten TLS/SSL ja DTLS riippuen protokollasta. Laitteet tulisi todentaa käyttämällä protokollan mukaan esimerkiksi SASL-viitekehyksen salausmekanismeja, käyttäjätunnusta ja salasanaa, tai sertifikaatteja. Tietoturvaa voidaan parantaa myös pitämällä laitteiden ohjelmisto ajan tasalla ja käyttämällä oletusarvoisten käyttäjätunnusten sekä salasanojen sijasta vahvoja salasanoja.

MQTT-protokolla on sovelluskerroksen protokolla, jota voidaan käyttää lukuisissa eri sovelluksissa, kuten koti- ja teollisessa automaatiossa tai laitteiden välisessä viestinnässä. Protokolla tukee TLS/SSL-salausprotokollia yhteyden suojaamiseksi sekä tietojen

eheyden varmistavia virheenhavaitsemismenetelmiä. Tietoturvaa ei kuitenkaan ole MQTT:n oletusasetuksilla, vaan protokolla tulee konfiguroida asianmukaisesti tietoturvan toteuttamiseksi.

CoAP on sovelluskerroksen protokolla ja se tunnetaan lähinnä viestintäprotokollana hyvin resurssirajoitetuissa laitteissa sekä verkoissa. CoAP-protokolla tukee DTLS-suojausta, joka mahdollistaa monta eri suojaustilaa. Tämän lisäksi OSCORE-salausprotokollaa voidaan käyttää matalaa viestinnällistä kuormitusta priorisoivaan salaukseen, kun taas DTLS-suojaus lisää viestinnällistä kuormitusta. Kuten MQTT-protokollassa, myös CoAP-protokollassa tulee konfiguroida tietoturvaominaisuudet asianmukaisella tavalla.

XMPP on avoimen standardin sovelluskerroksen protokolla, joka soveltuu käytettäväksi reaaliaikaiseen viestintään ja sellaisiin IoT-sovelluksiin, joita halutaan laajentaa lisätoiminnallisuuksilla. XMPP käyttää TLS-salausprotokollaa yhteyden suojaamiseksi ja SASL-viitekehyksen mekanismeja laitteiden todentamiseksi. XMPP-protokollan päästä päähän -suojaus ei välttämättä toteudu aina, sillä viestintä koostuu useista XML-tietovirroista, jotka eivät ole aina TLS-suojattuja. Päästä päähän -suojaus voidaan varmistaa OMEMO-laajennuksella.

Sovelluskerroksen protokollaa AMQP voidaan käyttää muun muassa liiketoiminta- ja kaupallisissa IoT-sovelluksissa. Se ei kuitenkaan sovellu reaaliaikaiseen viestintään, kuten muut sovelluskerroksen protokollat, ja kuluttaa enemmän virtaa verrattuna MQTT-protokollaan. AMQP-verkkoon voidaan määrittää TLS-salausprotokolla ja SASL-viitekehyksen mekanismit yhteyden suojaamiseksi sekä laitteiden todentamiseksi. TLS-suojaus on kuitenkin purettavissa riittävillä resursseilla. AMQP-protokolla on myös haavoittuvainen, sillä siihen voidaan avoimen lähdekoodin myötä lisätä tietoturvariskejä lisääviä lisätoiminnallisuuksia.

Taulukossa (Taulukko 1) on esitetty työssä käsiteltyjen sovelluskerroksen protokollien viestintämallit, ominaisuudet sekä oleelliset tietoturvaominaisuudet.

Taulukko 1. Sovelluskerroksen protokollien ominaisuuksia

Protokolla	Viestintämalli	Ominaisuudet	Tietoturvaominaisuudet
MQTT	-julkaisija/tilaaja	-alhainen virrankulutus -suunniteltu tehokkaaseen viestintään -käyttää TCP-protokollaa	-TLS/SSL-suojaus -todentaminen käyttäjätunnuksella ja salasanalla
CoAP	-pyyntö/vastaus -REST-arkkitehtuuri	-alhainen virrankulutus -toimii resurssirajoitetuissa verkoissa ja laitteissa -soveltuu koneiden väliseen viestintään (M2M) -käyttää UDP-protokollaa	-DTLS-suojaus -neljä eri suojaustilaa (DTLS) -OSCORE-suojaus
XMPP	-julkaisija/tilaaja -pyyntö/vastaus -asynkroninen viestintä	-keskitason virrankulutus -perustuu XML-kieleen -reaaliaikainen viestintä -laajennettavuus -käyttää TCP-protokollaa	-TLS/SSL-suojaus -SASL-viitekehys asiakkaiden todentamiseksi -päästä päähän -suojaus (OMEMO)
AMQP	-julkaisija/tilaaja (aiheet) -pisteestä pisteeseen -jonot	-keskitason virrankulutus -luotettava viestintä -laajennettavuus -käyttää TCP-protokollaa	-TLS/SSL-suojaus -SASL-viitekehys asiakkaiden todentamiseksi

LoRaWAN on LoRa-tekniikkaa käyttävä siirtokerroksen MAC-alikerroksen protokolla, joka on suunniteltu yhdistämään paristokäyttöisiä IoT-laitteita langattomasti jopa 15

kilometrin kantamalla. Sitä käytetään muun muassa maataloudessa, rakennuksissa sekä teollisuudessa. LoRaWAN käyttää tietoturvaominaisuuksien toteuttamiseen standardisoituja algoritmeja. Protokollassa toteutuu tietojen luottamuksellisuus ja eheys käyttämällä erilaisia salausavaimia sekä MIC-koodia viestien peukaloinnin havaitsemiseksi. LoRaWAN on altis hyökkäyksille, joissa pyritään pääsemään käsiksi esimerkiksi fyysisiin laitteisiin. Teoriassa myös MIC-koodi voidaan murtaa, jolloin tietojen eheys saattaisi vaarantua.

Zigbee-protokolla soveltuu verkkokerroksen protokollana IoT-sovelluksiin, joissa vaaditaan luotettavaa ja alhaisen virrankulutuksen tiedonsiirtoa pienillä tiedonsiirtonopeuksilla. Zigbee käyttää AES-128-standardia tietojen salaamiseen, tietojen eheyden varmistamiseksi sekä laitteiden todentamiseksi. Tietojen eheys varmistetaan vielä MIC-koodilla. Zigbee on haavoittuvainen erilaisille hyökkäyksille, kuten väliintulo- ja palvelunestohyökkäyksille. Haavoittuvuudet voidaan kuitenkin korjata työssä esitetyillä tavoilla.

NB-IoT on jo olemassa oleviin standardeihin perustuva esineiden internetiä varten suunniteltu LPWA-teknologia. NB-IoT mahdollistaa pitkän akunkeston IoT-laitteille ja toimii optimoidusti rinnakkain mobiiliverkkojen kanssa ilman tarvetta uudelle infrastruktuurille. NB-IoT:n tietoturva toteutetaan osittain jo olemassa olevien mobiiliverkkojen tarjoamilla tietoturvaominaisuuksilla. Tietoturvaa voidaan lisätä vaihtoehtoisilla NB-IoT-ominaisuuksilla, kuten TLS-suojauksella tai BEST-ominaisuudella.

Thread on esineiden internetiä varten suunniteltu protokolla, jota käytetään lähinnä kotiautomaatiossa ja kaupallisissa sovelluksissa, joissa halutaan käyttää IP-protokollaa. Thread perustuu olemassa oleviin avoimiin standardeihin, joten myös Threadin tietoturvaominaisuudet riippuvat osittain sen käytössä avointen standardien tietoturvaominaisuuksista. Threadiin on vuosien saatossa lisätty tietoturvaominaisuuksia, kuten todennettu TLS-istunto laitteen käyttöönottovaiheessa. Threadia päivitetään jatkuvasti, ja sen uusin versio on julkaistu vuonna 2024.

Taulukossa (Taulukko 2) on esitetty protokollien LoRaWAN, Zigbee, NB-IoT ja Thread IoT-protokollapinon kerrokset, ominaisuudet sekä tietoturvaominaisuudet.

Taulukko 2. LoRaWAN, NB-IoT, Zigbee ja Thread ominaisuudet ja tietoturvaominaisuudet

Protokolla	Kerros	Ominaisuudet	Tietoturvaominaisuudet
LoRaWAN	Siirtokerros (MAC)	-pitkä kantama -alhainen virrankulutus -langaton yhteys paristokäyttöisille laitteille	-AES-128-standardin salausavaimet -MIC-koodi -päästä päähän -suojaus
Zigbee	Sovellus/verkkokerros	-luotettava viestintä -pienet tiedonsiirtonopeudet	-AES-128-standardin salausavaimet -MIC-koodi -turvakeskus avaintenhallintaan -päästä päähän -suojaus
NB-IoT	Verkkokerros	-käyttää olemassa olevia standardeja ja infrastruktuuria -laitteiden pitkä akunkesto	-käyttää mobiiliverkkojen tietoturvaominaisuuksia -BEST-ominaisuus -TLS-suojaus
Thread	Verkkokerros	-käyttää olemassa olevia standardeja -mesh-topologia lisää viestinnän luotettavuutta -suunniteltu erityisesti IoT varten	-perustuu osittain sen käyttämien standardien tietoturvaominaisuuksiin -TLS-suojaus laitteen käyttöönottovaiheessa

Työssä käsiteltyjä protokollia on käytetty IoT-laitteissa ja -sovelluksissa jo pitkään, ja osaa protokollista päivitetään edelleen. Tulevaisuuden osalta erityisesti Thread-protokollan käytön odotetaan lisääntyvän, sillä kyseistä protokollaa päivitetään usein, ja se on päivitysten myötä tietoturvallinen vaihtoehto moneen eri käyttökohteeseen esineiden internetissä.

Tutkielman aihe on laaja, eikä työssä voinut mitenkään käsitellä kaikkia esineiden internetissä käytettäviä protokollia, eikä myöskään kaikkia käsiteltyjen protokollien ominaisuuksia. IoT:n tietoturvan merkitys kasvaa merkittävästi jatkuvasti, joten tutkimusta voitaisiin jatkaa selvittämällä protokollien haavoittuvuuksia ja parempia ratkaisuja tietoturvaominaisuuksien toteuttamiseksi.

Lähteet

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayaash, M. *Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications*. IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2347-2376. https://doi.org/10.1109/COMST.2015.2444095

Alahari, H. P., & Yalavarthi, S. B. (2017). *A survey on network routing protocols in internet of things (IOT).* International Journal of Computer Applications, 160(2), 18-22. https://doi.org/10.5120/ijca2017912973

Ashton, K. (2009). *That 'Internet of Things' thing.* RFID Journal. Noudettu 18.10.2024. osoitteesta https://www.rfidjournal.com/expert-views/that-internet-of-things-thing/73881/

Cloudflare. (n.d.-a). *What is a protocol?* Cloudflare. Noudettu 13.10.2024. osoitteesta https://www.cloudflare.com/learning/network-layer/what-is-a-protocol/

Cloudflare. (n.d.-b). *What is TLS (Transport Layer Security)?* Cloudflare. Noudettu 18.10.2024. osoitteesta https://www.cloudflare.com/learning/ssl/transport-layer-security-tls/

Cloudflare. (n.d.-c). *What is SSL? | SSL definition*. Cloudflare. Noudettu 18.10.2024. osoitteesta https://www.cloudflare.com/learning/ssl/what-is-ssl/

Coman, F. L., Malarski, K. M., Petersen, M. N., & Ruepp, S. (2019). Security issues in Internet of Things: Vulnerability analysis of LoRaWAN, Sigfox, and NB-IoT. In *2019 Global IoT Summit (GIoTS)* (pp. 1-6). Aarhus, Tanska. https://doi.org/10.1109/GIOTS.2019.8766430

Gerber, A., Romeo, J. (2020). *Connecting all the things in the Internet of Things.* IBM

Developer. Noudettu 20.10.2024. osoitteesta https://developer.ibm.com/articles/iot-lp101-connectivity-network-protocols/

Gemalto, Actility, & Semtech. (2017). *LoRaWAN™ Security : Full end-to-end encryption for IoT application providers*. LoRa Alliance. Noudettu 25.11.2024. osoitteesta https://lora-alliance.org/wp-content/uploads/2020/11/lorawan_security_whitepaper.pdf

GSM Association (n.d.). *Narrowband – Internet of Things (NB-IoT).* Noudettu 5.1.2025. osoitteesta https://www.gsma.com/solutions-and-impact/technologies/internet-of-things/narrow-band-internet-of-things-nb-iot/

GSM Association (2019). NB-IoT Deployment Guide to Basic Feature Set Requirements. Noudettu 5.1.2025. osoitteesta https://www.gsma.com/solutions-and-impact/technologies/internet-of-things/wp-content/uploads/2019/07/201906-GSMA-NB-IoT-Deployment-Guide-v3.pdf

Hernández Ramos, S., Villalba, M. T., & Lacuesta, R. (2018). *MQTT security: A novel fuzzing approach. Wireless Communications and Mobile Computing*. https://doi.org/10.1155/2018/8261746

HiveMQ Team. (2024a, 6. maaliskuuta). *TLS/SSL - MQTT security fundamentals*. HiveMQ. Noudettu 18.10.2024 osoitteesta https://www.hivemq.com/blog/mqtt-security-fundamentals-tls-ssl/

HiveMQ Team. (2024b, 14. helmikuuta). *Introducing the MQTT Protocol – MQTT Essentials: Part 1*. HiveMQ Team. Noudettu 18.10.2024. osoitteesta https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/

Kaskatiiski, N., & Boyanov, L. (2021). Efficiency of data exchange of IoT communication

protocols. *2021 International Conference Automatics and Informatics (ICAI).* Varna, Bulgaria, (pp. 358-361). https://doi.org/10.1109/ICAI52893.2021.9639627

Khanji, S., Iqbal, F., Hung, P. (2019). *ZigBee Security Vulnerabilities: Exploration and Evaluating*. 2019 10th International Conference on Information and Communication Systems (ICICS), Irbid, Jordan, (pp. 52-57). https://doi.org/10.1109/IACS.2019.8809115

LoRa Alliance. (n.d.). *What is LoRaWAN?* LoRa Alliance. Noudettu 25.11.2024. osoitteesta https://lora-alliance.org/about-lorawan/

Malik, M. I., McAteer, I. N., Hannay, P., Syed, N. F., & Zubair, B. (2018). XMPP architecture and security challenges in an IoT ecosystem. In *Proceedings of the 16th Australian Information Security Management Conference* (pp. 62-73). Edith Cowan University, Perth, Australia. https://doi.org/10.25958/5c52735166690

McAteer, I., Malik, M. I., Baig, Z., & Hannay, P. (2017). Security vulnerabilities and cyber threat analysis of the AMQP protocol for the Internet of Things. In C. Valli (Ed.), *The Proceedings of the 15th Australian Information Security Management Conference* (pp. 70-80). Edith Cowan University, Perth, Western Australia. https://doi.org/10.4225/75/5a84f4a695b4c

O'Hara, John. (2007). *Toward a Commodity Enterprise Middleware.* https://doi.org/10.1145/1255421.1255424

OASIS. (2012). *OASIS Advanced Message Queuing Protocol (AMQP) Version 1.0.* OASIS Standard. Noudettu 1.12.2024 osoitteesta https://docs.oasis-open.org/amqp/core/v1.0/amqp-core-complete-v1.0.pdf

Postel, J. (1980). *User Datagram Protocol (UDP)*. RFC 768. Internet Engineering Task

Force. Noudettu 21.10.2024. osoitteesta https://datatracker.ietf.org/doc/html/rfc768

Postel, J. (1981). *Transmission Control Protocol (TCP)*. RFC 793. Internet Engineering Task Force. Noudettu 21.10.2024. osoitteesta https://datatracker.ietf.org/doc/html/rfc793

Rahman, R. A., & Shah, B. (2016). Security analysis of IoT protocols: A focus on CoAP. In *2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC)* (pp. 1-7). Muscat, Oman. https://doi.org/10.1109/ICBDSC.2016.7460363

Rescorla, E., & Modadugu, N. (2012). *Datagram Transport Layer Security Version 1.2.* RFC 6347. Internet Engineering Task Force. Noudettu 20.11.2024. osoitteesta https://datatracker.ietf.org/doc/html/rfc6347

Saint-Andre, P. (2011). *Extensible Messaging and Presence Protocol (XMPP): Core.* RFC 6120. Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/rfc6120

Selander, G., Mattsson, J., Palombini, F., & Seitz, L. (2019). *Object Security for Constrained RESTful Environments (OSCORE).* RFC 8613. Internet Engineering Task Force. Noudettu 20.11.2024 osoitteesta https://datatracker.ietf.org/doc/html/rfc8613

Semtech (n.d.). *What is LoRa?* Semtech. Noudettu 25.11.2024. osoitteesta https://www.semtech.com/lora/what-is-lora

Sharma, C., & Gondhi, N. K. (2018). Communication Protocol Stack for Constrained IoT Systems. *2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU)*. Bhimtal, Intia. (pp. 1-6). https://doi.org/10.1109/IoT-SIU.2018.8519904

Shelby, Z., Hartke, K., Bormann, C. (2014). *The Constrained Application Protocol (CoAP).* RFC 7252. Internet Engineering Task Force. Noudettu 14.10.2024. osoitteesta https://datatracker.ietf.org/doc/html/rfc7252

Suleymanov, E., Kirdan, E., & Pahl, M. -O. Securing CoAP with DTLS and OSCORE. *2022 6th Cyber Security in Networking Conference (CSNet)*, Rio de Janeiro, Brasilia. pp. 1-7. https://doi.org/10.1109/CSNet56116.2022.9955601

Swamy, S. N., Jadhav D., & Kulkarni N. (2017). Security threats in the application layer in IOT applications. *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*. Palladam, India. pp. 477-480. https://doi.org/10.1109/I-SMAC.2017.8058395

Thread Group. (2022). *Thread Network Fundamentals White Paper*. Noudettu 10.1.2025. osoitteesta https://portal.threadgroup.org/DesktopModules/Inventures_Document/FileDownload.aspx?ContentID=633

Thread Group. (2024). *Thread 1.4 Features White Paper*. Noudettu 10.1.2025. osoitteesta https://www.threadgroup.org/Portals/0/Documents/Thread_1.4_Features_White_Paper_September_2024.pdf

Tomić, I., & McCann, J. A. (2017). *A Survey of Potential Security Issues in Existing Wireless Sensor Network Protocols*. IEEE Internet of Things Journal, vol. 4, no. 6, pp. 1910-1923. https://doi.org/10.1109/JIOT.2017.2749883

Unwala, I., Taqvi, Z., & Lu, J. (2018). Thread: An IoT protocol. *2018 IEEE Green*

Technologies Conference (GreenTech) (pp. 161-167). Austin, TX, USA. https://doi.org/10.1109/GreenTech.2018.00037

W3Schools. (n.d.). *Introduction to XML.* Noudettu 27.11.2024. osoitteesta https://www.w3schools.com/xml/xml_whatis.asp

XMPP. (n.d.-a). *An Overview of XMPP.* Noudettu 27.11.2024. osoitteesta https://xmpp.org/about/technology-overview/

XMPP. (n.d.-b). *Internet of Things (IoT).* Noudettu 27.11.2024. osoitteesta https://xmpp.org/uses/internet-of-things/

XMPP. (n.d.-c). Specifications. Noudettu 27.11.2024. osoitteesta https://xmpp.org/extensions/

XMPP Standards Foundation. (2022). *XEP-0384: OMEMO Encryption*. Noudettu 27.11.2024. osoitteesta https://xmpp.org/extensions/xep-0384.html

Zigbee Alliance. (2015). *Zigbee Specification.* Noudettu 25.11.2024. osoitteesta https://zigbeealliance.org/wp-content/uploads/2019/11/docs-05-3474-21-0csg-zigbee-specification.pdf