Kauno technologijos universitetas

Informatikos fakultetas

Daiktų interneto MQTT protokolo saugos ir energijos sąnaudų tyrimas

Baigiamasis magistro profesinių studijų projektas

Edgaras Baranauskas

Projekto autorius

prof. Jevgenijus Toldinas

Vadovas

Kaunas, 2019

Kauno technologijos universitetas
Informatikos fakultetas

Daiktų interneto MQTT protokolo saugos ir energijos sąnaudų tyrimas

Baigiamasis magistro profesinių studijų projektas
Informacijos ir informacinių technologijų sauga (6211BX008)

Edgaras Baranauskas
Projekto autorius

prof. Jevgenijus Toldinas
Vadovas

dr. Ignas Martišius
Recenzentas

Kaunas, 2019

Kauno technologijos universitetas

Informatikos fakultetas

Edgaras Baranauskas

Daiktų interneto MQTT protokolo saugos ir energijos sąnaudų tyrimas

Akademinio sąžiningumo deklaracija

Patvirtinu, kad mano, Edgaras Baranauskas, baigiamasis projektas tema „Daiktų interneto MQTT protokolo saugos ir energijos sąnaudų tyrimas“ yra parašytas visiškai savarankiškai ir visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

(vardą ir pavardę įrašyti ranka)	(parašas)

Baranauskas, E. „Daiktų interneto MQTT protokolo saugos ir energijos sąnaudų tyrimas“. Magistro profesinių studijų baigiamasis projektas / vadovas prof. Jevgenijus Toldinas; Kauno technologijos universitetas, Informatikos fakultetas.

Kaunas, 2019. 56 p.

SANTRAUKA

Daiktų internetas - interneto dalis, kuri apibrėžiama kaip globali ir dinaminė tinklo infrastruktūra, gebanti save konfigūruoti, pagrįsta standartiniais ir suderinamais komunikavimo protokolais, kur fiziniai ir virtualūs objektai turi identifikacines savybes, fizinius atributus, virtualias personalijas, naudoja išmaniąsias sąsajas ir tiesiogiai integruoti į vientisą informacinį tinklą. Kadangi integravimas ir apjungimas į tinklą įvairių daiktų interneto įrenginių yra neišvengiamas, todėl reikia padaryti taip, kad tie daiktai kurie veikia mūsų kasdienybėje, veiktų saugiai. Saugumo ir privatumo klausimas taps viena iš vystymosi problemų, nes gali atsirasti didesnė baimė dėl galimų įsilaužimų ir verslo sutrikdymų. Taip pat šiai idėjai kelią skintis į priekį gali trukdyti ir technologinės galimybės, kurios nebūtinai leis norimą funkcionalumą. Prie to dar pridėkime nuoseklių standartų ir vieningos sistemos nebuvimą. Įvertinus šias priežastis galima nesunkiai suprasti, kodėl jau kelerius metus reklamuoto prekes automatiškai užsakančio šaldytuvo neturime savo namuose, todėl daiktų internetas pramonėje kol kas irgi yra lėtai į priekį judanti tendencija. Pagrindiniai DI požymiai:

- ***Sujungimas***: viskas gali būti sujungta su pasaulio informacijos ir ryšių infrastruktūra.
- ***Su daiktais susijusios paslaugos***: DI gali teikti su daiktais susijusias paslaugas, atsižvelgiant į tokius daiktų apribojimus kaip, pavyzdžiui, privatumo apsauga.
- ***Heterogeniškumas***: DI įtaisai yra nevienalyčiai, nes jie remiasi įvairiomis platformomis ir tinklais. Jie gali sąveikauti su kitais įtaisais arba paslaugų platformomis per skirtingus tinklus.
- ***Dinamiški pokyčiai***: įtaisų būsena keičiasi dinamiškai, pavyzdžiui, jie užmiega ir atsibunda, įjungiami ir / arba atjungiami, taip pat keičiasi jų vieta ir greitis. Be to, dinamiškai keistis gali ir įtaisų skaičius.
- ***Didžiulis mastas***: administruojamų ir tarpusavyje bendraujančių įtaisų skaičius bus bent 10 kartų didesnis nei prie dabartinio interneto prijungtų įtaisų skaičius. Dar svarbesnė galimybė tvarkyti generuojamus duomenis ir juos interpretuoti, siekiant valdyti įtaisus.
- ***Bendrosios saugumo galimybės***: autorizavimas, autentifikavimas, duomenų konfidencialumo ir vientisumo apsauga, privatumo apsauga, saugumo auditas ir antivirusinės priemonės.
- ***Specifinės saugumo galimybės***: jos yra glaudžiai susijusios su konkrečios taikomosios programos reikalavimais, pvz., mobiliųjų mokėjimų saugumo reikalavimais.

DI atsirado kaip interneto tolesnio vystymo rezultatas. Kaip ir kiekviena kuriama technologija, DI pirmiausia skirtas palengvinti ir pagerinti žmonių gyvenimą, todėl jo nauda neabejotina.

Magistrinio darbo objektas - daiktų interneto MQTT protokolo saugos ir energijos sąnaudų tyrimas. Šio darbo struktūra:

- Pirmojoje darbo dalyje pateikiama daiktų interneto, naudojamų protokolų analizė. Joje nagrinėjami DI protokolai jų struktūrą ir problematiką, paslaugų kokybės lygius, saugumą,

bei kylančios grėsmės šiuolaikinėje visuomenėje. Šioje darbo dalyje taipogi analizuojamas daiktų interneto saugumo politikų sudarymas ir pritaikymas.

- Antrojoje darbo dalyje pateikiamas daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistema. Jame apibrėžiama sistemos architektūra, abonento ir brokerio komunikacijų schemos ir sistemos veikimo principai. Įvairūs sistemoje vykstantys procesai yra iliustruoti grafikais. Taip pat šioje dalyje rasite pateiktą ir sistemos prototipą.
- Trečiojoje darbo dalyje pateikiami eksperimentinio tyrimo rezultatai. Eksperimento metu buvo ištirta MQTT protokolo įtampos kritimas naudojant skirtingus paslaugų kokybės lygius ir šifravimą (TLS).
- Darbo pabaigoje pateiktos išvados.

Baranauskas, E. Research of MQTT protocol security and energy consumption. Master's thesis / supervisor Assoc. Prof. Dr. Jevgenijus Toldinas; Department of Computer Science, Faculty of Informatics, Kaunas University of Technology. – Kaunas, 2019. – 56 p.

SUMMARY

The Internet of Things is a part of the Internet that is defined as a global and dynamic network infrastructure capable of self-configuration based on standard and compatible communications protocols, where physical and virtual objects have identification features, physical attributes, virtual personalities, smart interfaces, and direct integration into a seamless information network. Because integration and networking of various items of Internet devices is inevitable, it is necessary to do things that work in our daily lives safely. The issue of security and privacy will become one of the problems of development, as there may be greater fear of potential hacking and business disruption. Technological options that do not necessarily provide the desired functionality can also hinder this idea. Add to this the lack of consistent standards and a unified system. Having evaluated these reasons, one can easily understand why we have not been promoting our products in our own home for several years, so the Internet of Things in the industry is still a slow moving trend. Main features of IoT:

- Connectivity: everything can be combined with a global information and communication infrastructure.
- Object-related services: DIs may provide item related services, subject to restrictions on items such as privacy protection.
- Heterogeneity: DI devices are heterogeneous because they are based on different platforms and networks. They can interact with other devices or service platforms over different networks.
- Dynamic changes: Device status changes dynamically, such as falling asleep and waking, turning on and / or disconnecting, and changing their location and speed. In addition, the number of devices can change dynamically.
- Large scale: The number of devices that are administered and interconnected will be at least 10 times the number of devices connected to the current Internet. Even more important is the ability to process the generated data and interpret it to control devices.
- General security options: authorization, authentication, data confidentiality and integrity protection, privacy protection, security audit and anti-virus tools.
- Specific security options: they are closely related to specific application requirements, such as mobile payment security requirements.

IoT has emerged as a result of further development of the Internet. As with any technology that is being developed, IoT is primarily designed to facilitate and improve people's lives, so its benefits are beyond doubt.

The subject of the Master's thesis is the study of safety and energy costs of the Internet of Things MQTT protocol. Structure of this work:

- The first part of the work presents an analysis of the Internet of Things used, the protocols used. It examines DI's protocols, their structure and issues, service quality levels, security, and emerging threats in modern society. This part of the paper also analyzes the development and application of Internet of Things policies.

- The second part of the work presents the safety and energy cost profiling system of the Internet of Things devices using MQTT protocol for message exchange. It defines system architecture, subscriber and broker communication scheme, and system operation principles. Various processes in the system are illustrated in graphs. Also in this section you will find the system prototype as well.
- The third part of the work presents the results of the experimental study. During the experiment, the voltage drop of the MQTT protocol was analyzed using different levels of service quality and encryption (TLS).
- Conclusions at the end of the work.

Turinys

Lentelių sąrašas

Paveikslų sąrašas

Terminų ir santrumpų žodynas

IoT – Daiktų internetas (angl. *Internet of things*)

OSI – abstraktus ryšio protokolų, naudojamų ryšio ir kompiuteriuose tinkluose, aprašymas (angl. *Open Systems Interconnection Reference Model*)

P2P – taškas-taškas topologija (angl. *Point to point*)

WAMP - Žiniatinklio taikomųjų programų protokolas (angl. *Web Application Messaging Protocol*)

AMQP – Išplėstinio pranešimo eilės protokolas (angl. *Advanced Message Queuing Protocol*)

XMPP - Išplečiamas pranešimų ir buvimo protokolas (angl. *Extensible Messaging and Presence Protocol*)

D2D – veikimo principas: prietaisas su prietaisui (angl. device-to-device)

AMQP - Išplėstinio pranešimo eilės protokolas

6LoWPANs – interneto protokolo versija Nr. 6, veikianti negalinguose bevieliuose asmeniniuose tinkluose (angl. *IPv6 over Low-Power Wireless Personal Area Networks*)

AES – pažangus šifravimo standartas (šifravimo algoritmas) (angl. *Advanced Encryption Standard*)

CA – sertifikatų valdžia (išduodanti sertifikatus) (angl. *Certificate Authority*)

CoAP – ribojamųjų programų protokolas (angl. *Constrained Applications Protocol*)

CPU – centrinis procesorius (angl. *Central Processing Unit*)

DNS – sričių vardų sistema (angl. *Domain Name System*)

DoS – atsisakymo aptarnauti ataka (angl. *Denial of Service*)

DTLS - datagramų transporto sluoksnio apsauga (angl. *Datagram Transport Layer Security*)

HTTP – hipertekstų persiuntimo protokolas (angl. *Hypertext Transfer Protocol*)

ID – identifikatorius (angl. *Identifier*)

IoT – „daiktų internetas“ (angl. *Internet of Things*)

IPv6 – interneto protokolo versija Nr. 6 (angl. *Internet Protocol version 6*)

M2M – veikimo principas: prietaisas prietaisui (angl. *machine-to-machine*)

MD5 – žinučių santrauka Nr. 5 (maišos funkcija)

MQTT – žinučių eilės sudarymo telemetrijos transporto protokolas (angl. *Message Queuing Telemetry Transport Protocol*)

PSK – iš anksto pasidalintas raktas (angl. *Pre-Shared Key*)

QoS – paslaugų kokybės lygis (angl. *Quality of Service*)

RAM – laisvosios prieigos atmintis (angl. *Random Access Memory*)

RC4 – Rivest šifras Nr. 4 (srauto šifras) (angl. *Rivest Cipher 4*)

RSA – Rivest – Shamir – Adleman viešojo rakto kriptografinė sistema

SHA1 – saugus maišos algoritmas Nr. 1 (maišos funkcija) (angl. *Secure Hash Algorithm 1*)

SOC – „sistema ant lusto“ (angl. *System-on-a-Chip*)

SRAM – statinė laisvosios prieigos atmintis (angl. *Static Random-Access Memory*)

SSL / TLS – saugiųjų jungimų lygmens protokolas / transporto lygmens protokolas, skirtas saugumui užtikrinti TCP / IP tinkluose (angl. *Secure Socket Layer / Transport Layer Security*)

TCP / IP – transporto valdymo protokolas / interneto protokolas, t.y. protokolų rinkinys, aprašantis duomenų persiuntimą tarp įvairių tipų kompiuterių ir operacinių sistemų (angl. *Transport Control Protocol / Internet Protocol*)

UDP – vartotojo datagramų protokolas (angl. *User Datagram Protocol*)

USB – universali nuoseklioji kompiuterio magistralė (angl. *Universal Serial Bus*)

VPN – virtualusis privatusis tinklas (angl. *Virtual Private Network*)

WiFi – belaidis tinklas (angl. *wireless fidelity*)

XML – yra W3C rekomenduojama bendros paskirties duomenų struktūrų bei jų turinio aprašomoji kalba (angl. *Extensible Markup Language*)

Įvadas

Sąvoka „daiktų internetas“ apibrėžia intelektualius objektus, kurie yra tarpusavyje sujungti, naudojant įvairias tinklo sąsajas ir protokolus, pvz., riboto taikymo protokolą (angl. *Constrained Applications Protocol*), žinučių eilės sudarymo telemetrijos transporto protokolas MQTT, *MQTT-SN* (jutiklių tinklams), išplečiamas pranešimų ir buvimo protokolas XMPP, žiniatinklio taikomųjų programų protokolas WAMP, išplėstinio pranešimo eilės protokolas (angl. *Advanced Message Queuing Protocol*) ir daugelis kitų. Pagal „Gartner“ prognozes 2021 m. Visame pasaulyje DI (angl. *Internet of things*) galutinio saugumo sprendimai pasieks 631 milijoną dolerių [1]. Galima rasti įvairių šaltinių su nuomonėmis, kad iki 2020 metų, daiktų interneto įrenginių skaičius išaugs iki 50 milijardų [2]. Kitas šaltinis teigia, kad įrenginių skaičius iki 2025 metų pasieks 200 milijardų skaičių [3]. O štai dar vieno šaltinio spėjimu iki 2025 metų kiekvienas žmogus turėtų naudotis ne mažiau kaip devynis įrenginius [4].

Įrenginio sluoksnio (veikimo principas: prietaisas prietaisui (angl. *machine-to-machine*)) taikymo lygmeniu populiariausias yra *MQTT*, gerai žinomos debesų platformos, tokios kaip „*Amazon AWS*“, „*Microsoft Azure*“ ir „*IBM Watson*“ programėlių bendravimui tarpusavyje naudoja *MQTT* [5]. *MQTT* turi mažą atminties pėdsaką, mažą energijos suvartojimą ir geresnį informacijos paskirstymą gavėjams [6]. Dėl šios priežasties *MQTT* protokolas yra plačiai naudojamas veikimo principas: prietaisas su prietaisui ryšiams, kur vienas iš pagrindinių problemų yra užtikrinti įrenginių ir *D2D* ryšių saugumą [7]. *MQTT* turi tris paslaugų kokybės lygis ir daugiau jokių saugumo mechanizmų. Transporto sluoksnio saugumas (*TLS*) yra standartinis protokolas, esantis virš perdavimo valdymo protokolo (*TCP*), siekiant apsaugoti duomenis ryšiuose. *OASIS* [8] aiškiai rekomenduoja naudoti *TLS* kaip saugumo sprendimą transporto sluoksnyje. Todėl aš naudosiu populiariausią daiktų interneto protokolą kuris yra visai nesaugus, pritaikysiu jam saugą ir vertinsiu *MQTT* protokolo poveikį energijos suvartojimui naudojant paslaugų kokybės lygius per *TLS*. Taip pat atlikus vertinimą pasiūlysiu profiliavimo metodą.

Darbo tikslas – pasiūlyti daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistema. Ištirti daiktų interneto MQTT protokolo energijos sąnaudas įvairiuose servisų kokybės lygiose taikant SSL/TLS.

Darbo uždaviniai:

- Išanalizuoti esamus daiktų interneto protokolus;
- Išanalizuoti MQTT protokolo paslaugų kokybės lygius ir MQTT protokolo programinio lygmens karkasus;
- Pasiūlyti daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistemos modelį;
- Sukurti daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistemos prototipą;
- Atlikti MQTT protokolo saugos ir energijos sąnaudų profiliavimo sistemos prototipo eksperimentinį tyrimą, palyginti gautus rezultatus.

1. Daiktų interneto įrenginių duomenų apsikeitimo protokolai

Daiktų internetas šiomis dienomis labai sparčiai plintanti technologija, visus procesus norima automatizuoti, darbą kurį atlieka žmonės norima supaprastinti, o tam padeda daiktų internetas, įvairiausi prietaisai prijungti prie interneto, sensoriai ir panašūs dalykai, kurie vienaip ar kitaip yra susieję ir prijungti į internetą arba kitaip tariant į globalų tinklą. Kuo daugiau įrenginių, kuo daugiau protokolų, tuo sunkiau atlikti saugumo ir kontrolės vaidmenį šiuolaikinėje visuomenėje.

1.1. Daiktų interneto saugumo problematikos

Saugumo problemos ir grėsmės su kuriomis susiduria daiktų interneto įrenginiai:

- Fizinės atakos: Tokie išpuoliai yra sutelkti į IoT sistemos technines sudedamąsias dalis, o užpuolikas turi būti fiziškai arti IoT sistemos, kad išpuoliai veiktų. Keletas šių išpuolių:
    A. Mazgų šnipinėjimas: užpuolikas gali sugadinti jutiklio mazgą, fiziškai pakeisdamas visą mazgą ar jo dalį arba netgi elektroniniu būdu apklausdamas mazgus, kad gautų prieigą ir pakeistų slaptą informaciją, pvz., Bendrus kriptografinius raktus ar maršruto lenteles.
    B. Kenkėjiškų kodų injekcijos: užpuolikas sukompromituotas mazgas, fiziškai atliktų injekciją į jį su kenkėjišku kodu, kuris jam suteiktų prieigą prie IoT sistemos.
- Apgaulė: autentiškumas paskirstytoje aplinkoje yra labai sunkus, todėl kenkėjiški mazgai gali naudoti suklastotą tapatybę kenkėjiškų ar slaptų atakų atveju.
- Paslaugų sutrikdymas: užpuolikai išnaudoja kenkėjišku kodu užkrėstus mazgus ir riboje prieigą prie IoT sistemos.
- Peradresavimo atakos: tarpiniai kenkėjiški mazgai (pvz., WSN) gali keisti teisingus nukreipimo kelius duomenų rinkimo ir perdavimo metu.
- Duomenų tranzito išpuoliai: įvairūs išpuoliai prieš konfidencialumą ir vientisumą duomenų tranzito metu (pvz., šnipinėjimas ir žmogus viduryje).
- Duomenų nutekėjimas: užpuolikas gali lengvai pavogti duomenis žinodamas paslaugos ar taikymo pažeidžiamumą.

1.1 lentelė. Daiktų interneto grėsmių modelis

Sluoksnis	Pagrindinės grėsmės
Taikomasis lygmuo	Duomenų nutekinimas Paslaugų sutrikdymas Kenkėjiško kodo injekcija
Transporto lygmuo	Peradresavimo atakos Duomenų tranzito išpuoliai Paslaugų sutrikdymas
Fizinis lygmuo	Fizinės atakos Duomenų nutekinimas Peradresavimo atakos Duomenų tranzito išpuoliai

Taip atrodo grėsmių modelis vertinant skirtingas daiktų interneto grėsmes [9].

1.2. Daiktų interneto energijos sąnaudų problemos

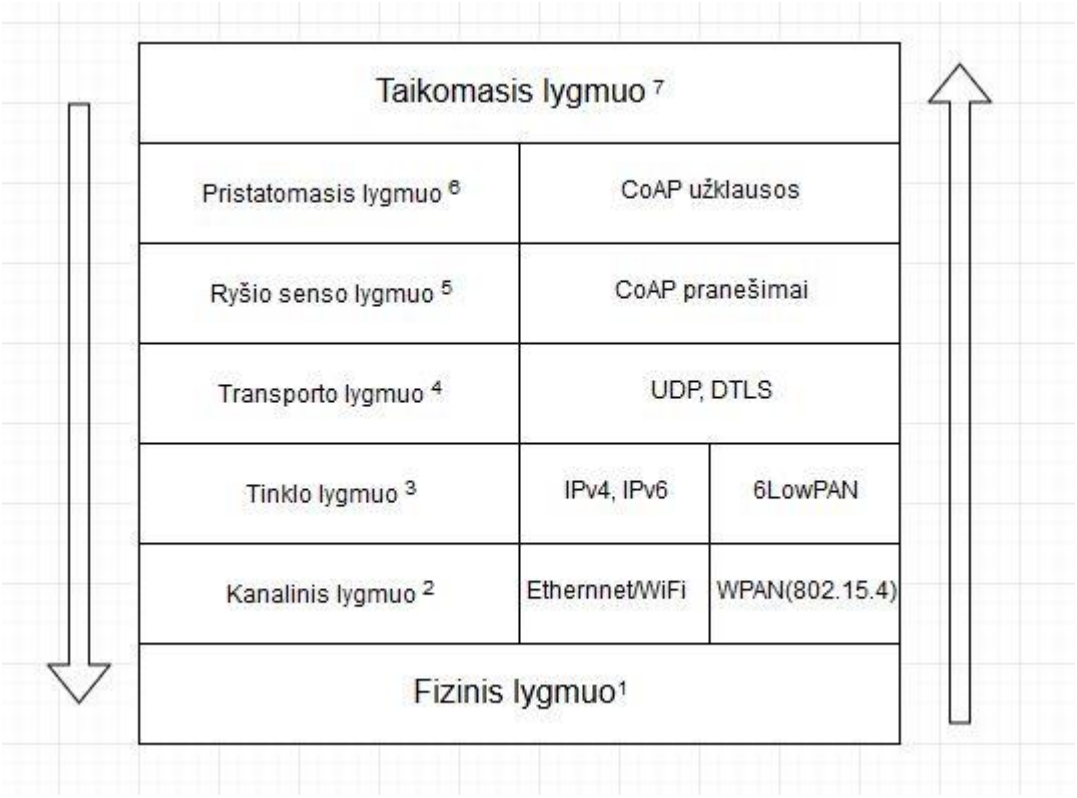
Išmaniųjų miestų, namų sprendimai turi būti energiją taupantys, ekonomiški, patikimi, saugūs, kad daiktų interneto įrenginiai galėtų veikti savarankiškai, užtikrinant paslaugų kokybės lygį, kad būtų pagerintas našumas nepertraukiamo tinklo veikimo metu. Todėl energijos vartojimo efektyvumas, daiktų interneto įrenginių eksploatavimo trukmė yra raktas į naujos kartos pažangius miesto sprendimus. Padidėjus išmaniųjų miestų daiktų interneto Programėlių naudojimui, energiją taupantys sprendimai, konkuruoja su mažos galios įrenginiais. Energiją taupantys sprendimai, pvz., lengvi protokolai, optimizavimo planavimas ir prognozuojantys energijos vartojimo modeliai, debesų sprendimai, mažos galios siųstuvai-imtuvai ir pažinimo valdymo sistema gali sumažinti energijos suvartojimą arba optimizuoti išteklių naudojimą. Galimos būsimos energijos valdymo kryptys išmaniuose miestuose yra [10]:

- Energiją taupantys mechanizmai programinės įrangos apibrėžtiems daiktų interneto sprendimams, kurie gali suteikti kintančio dydžio atitinkančius duomenis ir paslaugas;
- Kryptinis energijos perdavimas iš specialių energijos šaltinių bevielio maitinimo perdavimui;
- Energijos vartojimo efektyvumas ir saugumo protokolų sudėtingumas yra esminiai aspektai, susiję su jų praktiniu įgyvendinimu DI; taigi svarbu ištirti visus energijos apkrovimus, pritaikant saugumą daiktų interneto įrenginiams ir protokolams.

1.3. CoAP protokolas

Riboto taikymo protokolas (angl. *Constrained Applications Protocol*) yra specializuotas žiniatinklio perdavimo protokolas. Ribojamųjų programų protokolas CoAP, skirtas naudoti su tam tikrais mazgais ir tinklais kur yra ribotas pralaidumas ir energijos sąnaudos daiktų internete. Šis protokolas skirtas įrenginiams (M2M), pvz., pastatų automatizavimui. CoAP buvo suprojektuotas kaip lengvas protokolas, kuris gali veikti protinguose įrenginiuose, kuriuose trūksta atminties ir skaičiavimo išteklių. Supaprastintu paaiškinimu, CoAP yra labai panašus į HTTP (angl. *Hypertext Transfer Protocol*), bet užuot dirbęs ant TCP paketų, jis naudoja UDP paketus, lengvesnis duomenų perdavimo formatas, sukurtas kaip TCP alternatyva. Kaip ir HTTP yra naudojamas duomenims ir komandoms (GET, POST, CONNECT ir t.t.) tarp kliento ir serverio transportuoti, CoAP taip pat leidžia naudoti tas pačias komandų perdavimo funkcijas, tačiau nereikalaujant tokių pat išteklių, todėl jis idealiai tinka šiandienos kylančios daiktų interneto (IoT) įrenginių paklausai [11]. Tačiau, kaip ir bet kuris kitas UDP pagrįstas protokolas, CoAP iš esmės yra jautrus IP adresų sugadinimui ir paketų stiprinimui. Tai du pagrindiniai veiksniai, leidžiantys sustiprinti DDoS ataką. Užpuolikas gali siųsti nedidelį UDP paketą į CoAP klientą („IoT“ įrenginį), o klientas atsakytų daug didesniu paketu. DDoS atakų pasaulyje šio paketo atsako dydis yra žinomas kaip stiprinimo faktorius, o CoAP - tai nuo 10 iki 50, priklausomai nuo pradinio paketo ir gauto atsako (ir skaityto protokolo analizės). Be to, kadangi CoAP yra pažeidžiama pasiklausymo atakai, piktavalis gali pakeisti „siuntėjo IP adresą“ su nukentėjusiojo IP adresu, kuriam jie nori paleisti DDoS ataką, ir kad nukentėjusysis gautų nenuoseklią sustiprinto CoAP srauto jėgą.

Taip atrodytų CoAP OSI lygmenį:



1.1 pav. OSI lygmenys naudojant CoAP protokolą

Vienas iš svarbiausių CoAP projekto tikslų yra išvengti fragmentacijos pagrindiniuose sluoksniuose, ypač ryšių sluoksnyje, t.y. įmanoma naudojant kompaktišką 4 baitų binarinę antraštę, pasirinktinius laukus ir naudingąją apkrovą. Pranešimų tipas: CoAP palaiko keturių tipų pranešimus, kurių 2 bitai sandorių kodai.

CON	00(0)	ACK	10(2)
NON	01(1)	RST	11(3)

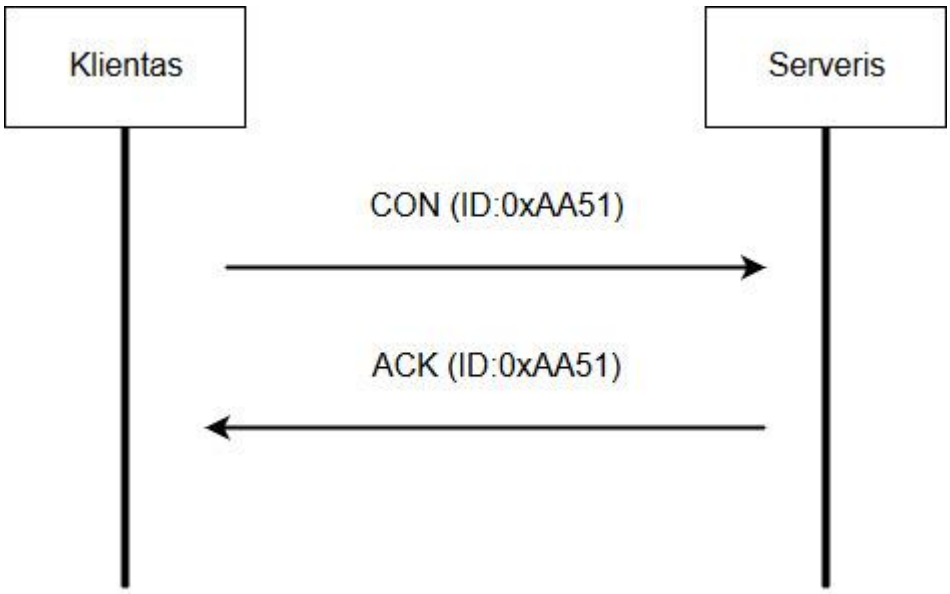
1.2 pav. CoAP 2 bitų pranešimai

CoAP siūlo neprivalomą patikimumą, naudojant suderinamus (CON) pranešimus, kai kiekvienas pranešimas (užklausa / atsakymas) turėtų būti patvirtintas (ACK) pagal pranešimų vertinimą. Pranešimai bus retransliuojami, jei CON žinutė per nustatytą laiką nebus patvirtinta. Atsakymą galima derinti su nepertraukiamu pripažinimu, kuris yra žinomas kaip „piggy“ palaikomas atsakas, arba atsakas gali būti išsiųstas po to, jei jis nėra nedelsiant prieinamas, o tai vadinama atskiru atsakymu. Nepagrįsti atsakymai į nepagrįstas užklausas turi būti patvirtinti, ir taip pat su netinkamais (NON) pranešimais [12].

CoAP palaiko keturis skirtingus pranešimų tipus:

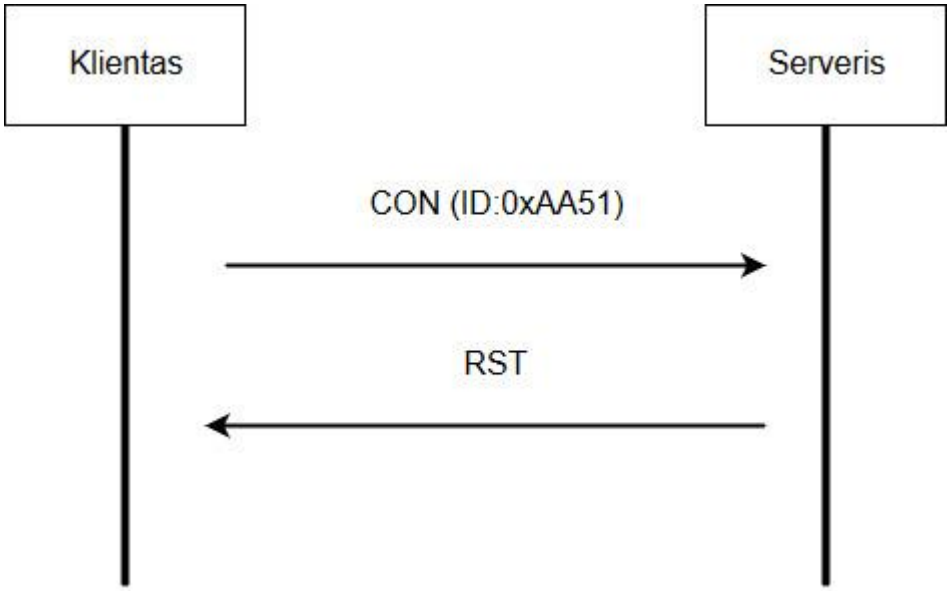
- Patvirtinamas pranešimas;
- Pranešimas be patvirtinimo;
- Pranešimo patvirtinimas;
- Pranešimo siuntimas iš naujo/pranešimo nutraukimas;

CoAP patvirtinamas pranešimas yra patikimas pranešimas. Keičiant pranešimus tarp dviejų taškų, šie pranešimai gali būti patikimi. „CoAP“ protokole gaunamas patikimas pranešimas naudojant patvirtinamąjį pranešimą (CON). Naudodamas tokį pranešimą, klientas gali būti tikras, kad pranešimas bus pasiekiamas serveryje. „CoAP Confirmable“ pranešimas siunčiamas dar kartą, kol kita šalis siunčia patvirtinimo pranešimą (ACK). ACK pranešime yra tas pats patvirtinamojo pranešimo ID (CON). Toliau pateiktame paveikslėlyje parodyta CoAP pranešimų mainų procedūra:



1.3 pav. Ryšys tarp kliento ir serverio su patvirtinimo pranešimu

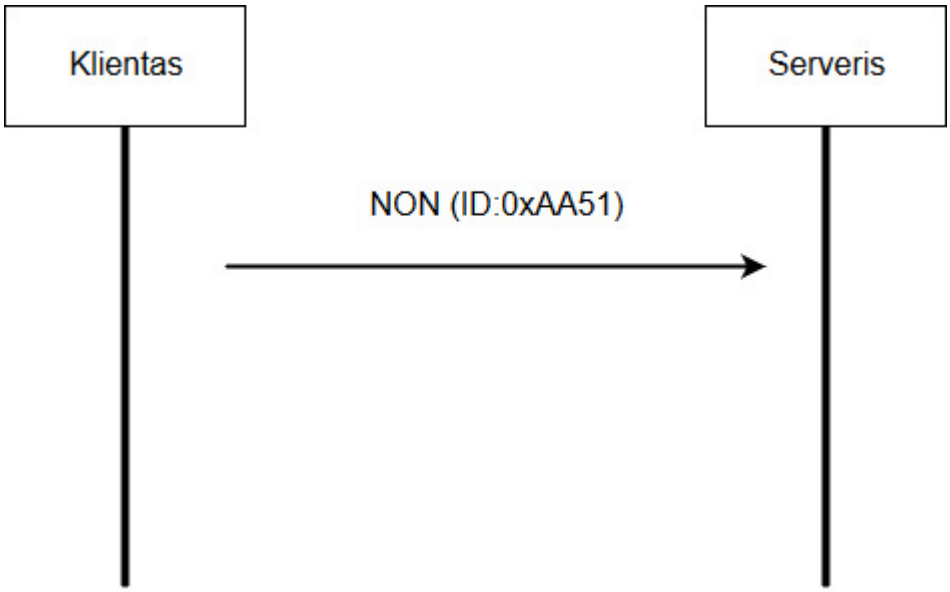
Jei serveryje yra problemų, susijusių su gaunamu prašymu, jis gali atsiųsti nutraukimo pranešimą (RST) vietoj patvirtinimo pranešimo (ACK).



1.4 pav. Ryšys tarp kliento ir serverio su nutraukimo pranešimu

Kita pranešimų kategorija yra nepatvirtinami (NON) pranešimai. Tai pranešimai, kuriems nereikia serverio patvirtinimo. Jie yra nepatikimi pranešimai arba, kitaip tariant, pranešimai, kuriuose

nėra kritinės informacijos, kuri turi būti pristatyta į serverį. Į šią kategoriją priskiriami pranešimai, turintys vertes, skaitomas iš jutiklių. Šie pranešimai yra nepatikimi, bet jie turi unikalų ID.



1.5 pav. Ryšys tarp kliento ir serverio su pranešimu be patvirtinimo

Vertinant CoAP protokolą per saugumo prizmę, jis išsiskiria tuo, kad pranešimams siųsti naudoja UDP paketą. O UDP paketo saugai užtikrinti yra naudojamas DTLS srauto šifravimas. Skirtingai nuo MQTT ar HTTP protokolų kuriuose duomenys yra persiunčiami naudojant TCP paketus ir TLS šifravima.

1.4. MQTT protokolas

MQTT yra žinučių eilės sudarymo telemetrijos transporto protokolas (angl. Message Queuing Telemetry Transport Protocol). Tai lengvas, atviro kodo, paprastas protokolas kuris suprojektuotas taip, kad jį būtų lengva įgyvendinti, bet kokioje infrastruktūroje. Šios charakteristikos idealiai tinka naudoti daugeliu atvejų, įskaitant ribotas aplinkas, tokias kaip komunikacija mašina su mašina (M2M) ir daiktų internete, kur reikalingi nedideli skaičiavimo resursai, tinklo pralaidumas, mažos energijos sąnaudos [13].

1999 m. MQTT buvo išradęs „*Andy Stanford-Clark*“ (IBM [14]) ir „*Arlen Nipper*“ ("*Arcom*", dabar "*Cirrus Link*"), kai jų naudojimo atvejis buvo sukurti protokolą dėl minimalaus baterijos praradimo ir minimalaus pralaidumo, jungiančio naftotiekius per palydovinį ryšį. Jie nurodė šiuos tikslus, kurie turėtų būti ateityje išspręsti:

- Paprasta įgyvendinti;
- Gera paslaugų duomenų kokybę;
- Lengvas ir efektyvus pralaidumas;
- Nuolatinis sesijos sąmoningumas;

Kaip ir bet kuris kitas interneto protokolas, MQTT yra pagrįstas klientais ir serveriu. Be to, serveris yra tas, kuris yra atsakingas už kliento prašymų gauti ar siųsti duomenis tarpusavyje tvarkymą. MQTT serveris vadinamas tarpininku, o klientai yra tiesiog prijungti įrenginiai [15]. MQTT pranešimų protokolai gali būti suskirstyti į kategorijas brokeriu/tarpininku pagrįstas protokolas, skirtas mašinoms

(M2M) komunikacija. Tarpininkas kontroliuoja platinimą informaciją. Klientai perkelia siuntėjo ir gavėjo vaidmenį, priklausomai nuo jų tikslų. MQTT teikia tris paslaugų kokybės lygius (QoS) 0, 1 ir 2. QoS yra pranešimo siuntėjo ir gavėjo susitarimas dėl pranešimo pateikimo garantijų. Remiantis neseniai atlikta apklausa, MQTT protokolas yra populiariausias tarp brokerio(tarpininko)/kliento žinučių perdavimo protokolų [16].

Saugumas MQTT yra padalintas į kelis sluoksnius. Kiekvienas sluoksnis apsaugo nuo įvairių rūšių atakų. Norint užtikrinti paslaugų saugumą ir patikimumą programėlėse kurias naudoja „Facebook“, reikalingas hibridinis MQTT protokolo pritaikymas ir įgyvendinimas. MQTT tikslas - suteikti lengvą ir lengvai naudojamą daiktų interneto ryšio protokolą. Gerai žinomos debesų platformos, tokios kaip „Amazon AWS“, „Microsoft Azure“ ir „IBM Watson“ taip pat naudoja MQTT. Šio protokolo įgyvendinimas ir diegimas paprastai naudoja kitus pažangiausius saugumo standartus: pvz., SSL / TLS transporto saugumui. Ypač integracija yra svarbus aspektas, nes dažnai brokeris yra komponentas, kuris tiesiogiai veikia žiniatinklį ir tvarko daug klientų, o vėliau siunčia pranešimus tolesniam analizės ir apdorojimo sistemoms. Apskritai tarpininkas yra centrinis mazgas, kuris kiekvieną žinutė turi perduoti. Čia yra aukšto lygio MQTT saugumo lygių santrauka:

- Tinklo lygmenį - vienas iš būdų užtikrinti saugų ir patikimą ryšį yra naudoti fiziškai saugų tinklą arba VPN visam klientų ir brokerių bendravimui;
- Transporto lygmenį - kai konfidencialumas yra pagrindinis tikslas, TLS / SSL paprastai naudojamas transporto šifravimui. Šis metodas yra saugus ir įrodytas būdas įsitikinti, kad duomenys negali būti perskaityti perdavimo metu, ir užtikrina kliento sertifikato autentiškumą, kad būtų galima patikrinti abiejų pusių tapatybę.
- Taikomajame lygmenį - Transporto lygmeniu ryšys yra užšifruotas ir tapatybės patvirtinamos. MQTT protokolas suteikia kliento identifikatorių ir naudotojo vardą / slaptažodį, kad būtų galima atpažinti įrenginius programų lygmeniu. Šios savybės pateikiamos pačiame protokole. Kiekvieno prietaiso leidimas arba kontrolė yra apibrėžta pagal konkretų tarpininko įgyvendinimą.

Reikalavimai autentiškumui, autorizacijai, duomenų vientisumui ir konfidencialumui neįtraukti į MQTT specifikaciją. Autoriai [17] teigia, kad saugumo reikalavimų trūkumas MQTT protokolo standarte yra susijęs su:

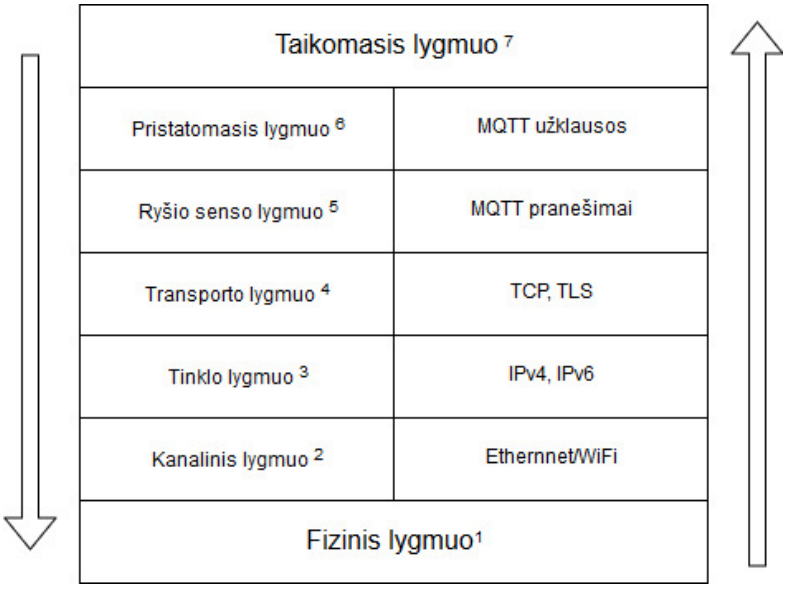
• MQTT orientuojasi tik į pranešimų siuntimą;

• Sumažinus pridėtinę vertę, susijusią su saugumo funkcijomis, protokolas naudojamas kiek įmanoma lengvesnis;

• MQTT istoriniai diegimai buvo pagrįsti privačiuose tinkluose;

• Kompleksiniai saugumo sprendimai reikalingi saugumui užtikrinti naudojant MQTT protokolą, kuris naudojamas „Facebook“, persiųsti pranešimus.

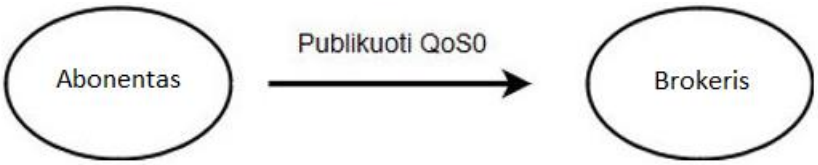
MQTT protokolas OSI lygmenų steke:



1.6 pav. MQTT protokolas OSI lygmenų steke

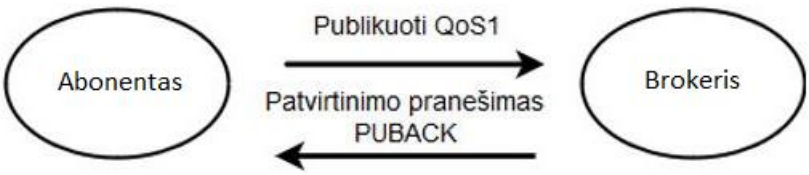
MQTT protokolas turi 3 paslaugų kokybės lygius:

- Daugiausiai vieną kartą (**QoS = 0**), kartais vadinamas „išsiųsti ir pamiršti“. Pranešimas yra siunčiamas daugiausiai vieną kartą arba visai nesiunčiamas. Jo pristatymas tinkle nepatvirtinamas jokiais papildomais pranešimais. Pranešimas nėra saugomas. Pranešimas gali būti prarastas, jei klientas atjungtas arba serveris yra nepasiekiamas.



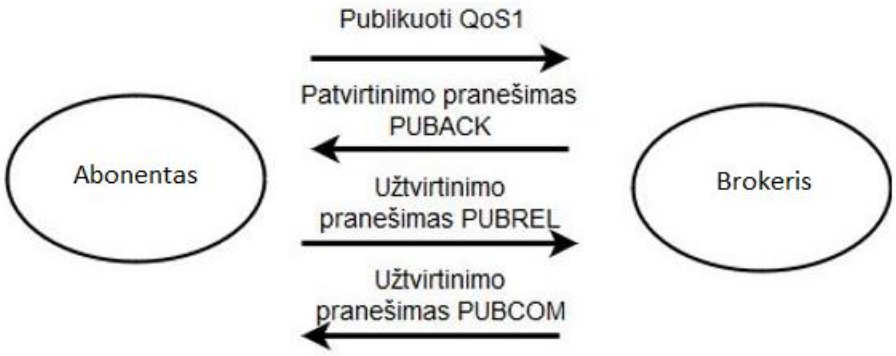
1.7 pav. Paslaugų kokybės lygis (QoS0)

- Bent kartą (**QoS = 1**) tai yra numatytasis persiuntimo būdas. Pranešimas visada pateikiamas bent kartą. Jei siuntėjas negauna patvirtinimo, pranešimas vėl išsiunčiamas su DUP vėliava, kol gaunamas patvirtinimas. Todėl gali būti išsiųstas tas pats pranešimas kelis kartus ir gali būti apdorotas kelis kartus. Pranešimas turi būti saugomas ir serverį ir pas klientą, tol kol jis bus apdorotas.



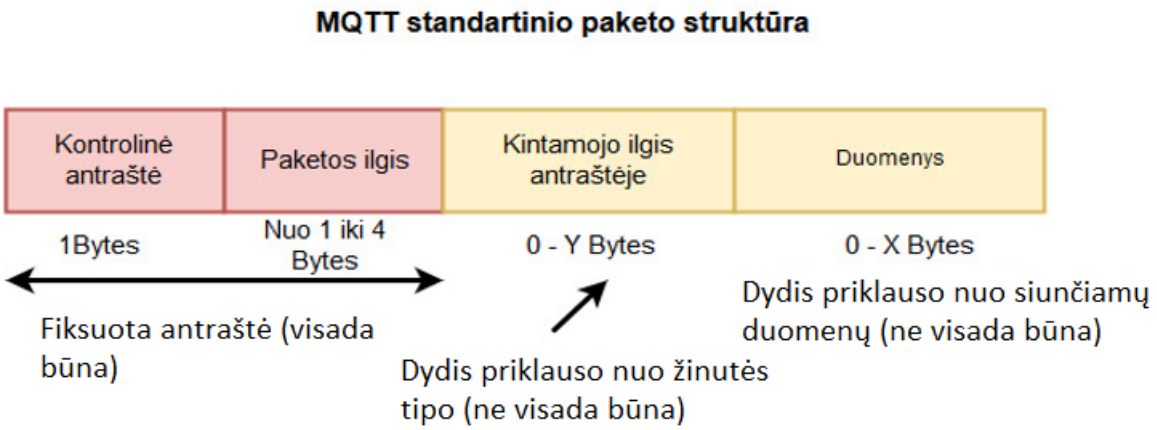
1.8 pav. Paslaugų kokybės lygis (QoS1)

- Tiksliai vieną kartą (**QoS = 2**) pranešimas visada pristatomas vieną kartą. Pranešimas turi būti saugomas serveryje ir pas klientą, kol jis bus apdorotas. QoS2 yra saugiausias, bet lėčiausias perdavimo būdas. Prieš išsiunčiant pranešimą iš serverio reikia patvirtinimo analogiškai ir kliento pusėje. Pranešimą gavėjas gali apdoroti po pirmojo perdavimo. Pirmoje siuntimo poroje siuntėjas perduoda pranešimą ir gauna patvirtinimą iš serverio, kad jis išsaugojo pranešimą. Jei siuntėjas negauna patvirtinimo, pranešimas vėl išsiunčiamas pranešimas su DUP vėliava, kol gaunamas patvirtinimas. Antroje perdavimo poroje siuntėjas praneša gavėjui, kad jis gali užbaigti pranešimo apdorojimą, PUBREL. Jei siuntėjas negauna PUBREL pranešimo patvirtinimo, PUBREL pranešimas vėl išsiunčiamas, kol bus gautas patvirtinimas. Siuntėjas ištrina išsaugotą pranešimą, kai gauna patvirtinimą į PUBREL pranešimą [18].



1.9 pav. Paslaugų kokybės lygis (QoS2)

MQTT paketo arba pranešimo formatas susideda iš 2 baitų fiksuotos antraštės (visada yra) + kintamos antraštės (ne visada yra) + siunčiami duomenys (ne visada yra). Taip atrodo standartinis MQTT paketas:



1.10 pav. Standartinio MQTT paketo struktūra

Fiksuotas antraštės laukas susideda iš valdymo lauko ir kintamo ilgio paketo ilgio laukų. Minimalus paketinio ilgio lauko dydis yra 1 baitas, skirtas pranešimams, kurių bendras ilgis yra

mažesnis nei 127 baitai (neįskaitant valdymo ir ilgio laukų). Maksimalus paketo dydis yra 256 MB. Mažesniuose nei 127 baitų paketuose yra 1 baito paketo ilgio laukas. Paketai, didesni nei 127 ir mažesni nei 16383, naudos 2 baitus. Naudojami 7 bitai, o 8-asis bitas yra tęsinys [19].

1.5. MQTT-SN protokolas

MQTT-SN - SN reiškia jutiklių/sensorių tinklą. MQTT eina per TCP / IP ir gali būti naudojamas lokaliam ryšiui arba veikti per internetą ar debesis. MQTT-SN gali būti naudojamas su daugiau protokolų, tinkamų jutikliams, tokiems kaip ZigBee, Z-Wave ir pan. Specifikacija skiriasi nuo MQTT skirtingai nei MQTT kuris veikia TCP / IP, MQTT-Sn veikia naudodamas UDP paketus. Tai dar lengvesnis protokolas nei MQTT, tačiau ir pritaikymas skiriasi, todėl ir naudojamas sensorių tinklams.

Pagrindinis MQTT-Sn privalumas yra paketų dydžio sumažinimas ir mažesnės susijungimo energijos sąnaudos, siekiant užtikrinti, kad pranešimo perdavimas suvartotų mažiau energijos. Pagrindiniai skirtumai tarp MQTT-SN ir MQTT yra šie [20]:

- WILL pranešimas yra neprivalomas ir gali būti siunčiamas kaip atskiras paketas. MQTT-SN CONNECT pranešimą padalijo į tris žinutes. Jūs tiesiog galite siųsti prisijungimo paketą tik tada, kai nenustatote „WILL“ temos ir pranešimo.
- Ilgą pranešimo pavadinimą nereikia siųsti su kiekvienu pranešimu. Vietoj to publikacijoje galima naudoti temos ID. Tai sumažina paketų dydį.
- Iš anksto nustatyti pranešimo pavadinimai padeda apibrėžti visus pranešimų išsiuntimo vartus. Pranešimų ID galima naudoti tiesiogiai. Lygiagretūs trumpi pranešimų pavadinimai, kurie yra mažesni nei 2 baitai, gali būti naudojami tiesiogiai be pranešimo ID.
- Automatinis klientų suradimas yra vienas didžiausių MQTT-SN privalumų. Klientams nereikia žinoti IP adreso arba „Gateway“ DNS pavadinimo arba tarpininko, kad prisijungtumėte ir pradėtumėte siųsti pranešimus.
- „Švari sesija“ išplėsta į „Will“ informaciją, kur bus išsami informacija, apie klientų prenumeruotus pranešimus.
- Jutikliai gali užmigti po to, kai pertam tikrą laiką negauna jokio pranešimo. Serveris siunčia buferio pranešimus, skirtus konkrečiam klientui, kai jis yra miego rėžime ir išsiųs atgal, kai klientas pabus gavęs tą pranešimą.

1.6. XMPP protokolas

XMPP (angl. *Extensible Messaging and Presence Protocol*) (anksčiau vadintas *Jabber*) – atviro standarto momentinių žinučių (pokalbių) internete sistema. XMPP yra pagrįstas XML pagrindu ir tai yra komunikacijos protokolas orientuotas į žinučių perdavimo protokolą. XMPP protokolas, yra atviro kodo kuris turi tarpines platformas žinutės perduoti. 1999 m. šį protokolą sukūrė „Jabber“ atviro kodo bendruomenė, skirta tiesioginiams momentiniams pranešimams persiųsti (IM). XMPP technologija susideda iš pagrindinių funkcijų ir daugelio kitų plėtinių, kuriuos galima integruoti su šia technologija, taip pat pati architektūra sukurta pranešimus siųsti/gauti beveik realiu laiku [21].

XMPP serveriai vieni kitus suranda pagal savo domeno vardą. Yra standartinis DNS SRV įrašas, kurį XMPP serveris siunčia užklausą, kad surastų, kuris serveris yra autoritetingas tam tikras domenas. Kai serveris bus identifikuotas, XMPP nustatys ryšį su serverio prievadu, nurodytu DNS. Tada jis patikrins nuotolinio serverio tapatybę naudodamas TLS ir sertifikatu pagrįsta autentifikavimu.

Šis autentifikavimas yra dvipusis - priimantis serveris taip pat patikrins tapatybę ir gaunamo serverio. Kadangi serveris yra patikimas, pranešimai iš serverio domeno, taip pat bus patikimi. Todėl XMPP architektūra tikrai gali būti laikoma panaši į SMTP. Kadangi SMTP yra gerai žinomas ir suprantamas pramonei, kuri leidžia dirbti su XMPP ir prisitaikyti prie jos.

XMPP naudojami servisai:

- Būsena (angl. *Presence*) - yra viena iš galingiausių XMPP servisų. Galimybė nustatyti, ar kitas vartotojas ar serveris yra prieinama ir jos esama būsena leidžia praktiškai keistis pranešimais beveik realiu laiku. Pasikeitus naudotojo būsenai, šis naujinimas bus pateikiamas realiu laiku visiems klientams ir serveriams.
- Įrašai/sąrašas (angl. *Roster*) - pateikiamas žinomi ir patvirtinti naudotojai, kad savininkas galėtų pamatyti kas šiuo metu yra aktyvūs kas ne ir galėtų tą įvertinęs pradėti bendrauti arba ne. Kada vartotojas prisijungia, jo būsena transliuojama į kiekvieno kontakto sąrašą. Šios paskyros atnaujins naudotojo būseną ir tada grąžins į esamą būseną.
- Pranešimai (angl. *Notifications*) - yra panašūs į individualius pranešimus, bet sistemą yra sukurta taip, kad daugumoje naujinimai būtų efektyviai nukreipti į vieną pusę. Panaši architektūrą kaip skelbti / prenumeruoti (angl. *Publish/Subscribe*) sistema, kurioje pranešimai siunčiami į skirtingus kanalus kartu su temomis. Tai leidžia klientams užsisakyti konkrečias juos dominančias temas ar pranešimus.
- Paslaugų atradimas (angl. *Service Discovery*) – leidžia klientams ir serveriams nustatyti paslaugų palaikymo funkcijas ir servisus, kitaip tariant ar abi pusės galės susitarti ir bendrauti naudojant tam tikrus servisus kurie veikia pas juo pačius.
- Suderinamumas (angl. *Jingle*) - yra XMPP plėtinys (XEP-0166) kuris leidžia XMPP išlikti paprasta technologija ir teikti paramą pažangiai integracijai su kitais protokolais ir sistemomis.

XMPP protokolo nauda:

- Atviro standarto (angl. *Open standart*) - XMPP protokolas IETF yra gerai dokumentuotas. Jis buvo įformintas RFC6120 ir RFC6121 standartizacijos procese, protokolas buvo kruopščiai peržiūrėtas ir išbandytas. Be to, kadangi protokolas yra atviro standarto, jis gali būti laisvai įgyvendinamas nesirūpinant dėl licencijavimo ar kitų kliūčių.
- Saugumas (angl. *Security*) - TLS šifravimas, tarp kliento ir serverio. Yra daug skirtingų serverių, klientų ir palaikančių bibliotekų diegimų, leidžiančių naudoti skirtingą kodo bazę. XMPP standartas vis dar vystosi, o papildomi saugumo patobulinimai yra vykdomi naujuose projektuose.
- Lankstus (angl. *Scalability*) – protokolas kuris gali būti keičiamo dydžio. Nėra nustatyta limito serveriams todėl jų galima įdiegti begales. Be to, XMPP naudoja „PUSH“ modelį.
- Realaus laiko (angl. *Real time*) - dėl „PUSH“ modelio XMPP vartotojai gali bendrauti realiu laiku. Kol priimantis agentas prisijungia prie savo serverio, jis gaus pranešimus, kai tik jis bus prieinami.

Be to ši technologija yra ganėtinai plačiai naudojama. Yra daug XMPP serverių ir klientų (programų ir bibliotekų), kuriuos jaus galima naudoti nekuriant naujų ir integruojant laiką į saugius sprendimus kuriuos galima naudoti su esama infrastruktūra [22].

1.7. WAMP protokolas

Žiniatinklio taikomųjų programų protokolas (angl. *Web Application Messaging Protocol*) yra maršrutinis protokolas, su visais komponentais jungiantis prie WAMP maršrutizatoriaus, kuris atlieka pranešimų nukreipimą tarp komponentų ir apjungia du žinučių siuntimo modelius viename žiniatinklio vietiniame protokole:

- Iššaukti nutolusias komandas (angl. *Remote Procedure Calls*)
- Skelbti / Prenumeruoti (angl. *Publish / Subscribe*)

Nemokamas atviro kodo protokolas kuri gali būti naudojamas ir diegiamas į įvairias sistemas, kur reikalaujama tokio pat mažo resursų naudojimo kaip ir aukščiau išvardintuose protokoluose. Kadangi protokolas apjungia du žinučių modelius todėl jam teikiami šie privalumai:

- komponentų, kurie nėra pasiekiami iš išorės tinklo lygmenį, iškviesti tam tikras funkcijas (pvz., (angl. *network address translation*)), bet kurie gali sukurti išeinančio tinklo ryšį su WAMP maršrutizatoriumi
- OSI lygmenų atsiejimas transporto lygmens nuo taikomojo lygmens, leidžia siųsti atvirkštines komandas, kai debesimis grindžiama sistema gali saugiai valdyti nuotolinius įrenginius
- Kadangi šiam protokolo darbui (abiem kryptimis) nereikia atidaryti jokių prievadų, įrenginių nuotolinio atakavimo paviršius yra visiškai uždarytas pvz. DOS atakoms

Sujungus skelbimo ir prenumeratos bei nuotolinių komandų iškvietimą viename žiniatinklio vietiniame realaus laiko transportavimo protokole, galima naudoti visoms komponentų ir mikroservisų programų pranešimų siuntimo reikmėms, mažinant technologinį sudėtingumą ir pridėtines sąnaudas, užtikrinant saugą.

1.8. Daiktų interneto įrenginių bendravimo protokolų apibendrinimas

Didėjant daiktų interneto populiarumui ir augant pačių įrenginių skaičiui. Didėja ir protokolų skaičius, kuriuos galima vienu ar kitu atveju pritaikyti savo poreikiams patenkinti. Palyginimui pasirinkau tris kriterijus pagal kuriuos palyginsiu išnagrinėtus protokolus, tai yra:

- Sauga
- Paslaugų kokybės lygis
- Energija

1.2 lentelė. Daiktų interneto protokolų palyginimas pagal pasirinktus kriterijus

IoT protokolas	Sauga	Paslaugų kokybės lygis
CoAP	DTLS	QoS žemas, QoS aukštas
MQTT	SSL/TLS	QoS0, QoS1, QoS2
MQTT-SN	DTLS	QoS0, QoS1, QoS2, QoS3
XMPP	TLS	-
WAMP	SSL	-

1.9. Analizės išvados

- Išanalizavus daiktų interneto protokolus, nustatyta, kad šiuo metu vienas iš plačiausiai naudojamų rinkoje yra MQTT protokolas, jis naudojamas tokių didžiųjų kompanijų kaip „Facebook“, „IBM“.
- Išanalizavus daiktu interneto protokolus nustatyta, kad MQTT protokolas charakterizuojamas mažomis energijos ir kitų resursų sąnaudomis.
- MQTT protokolas turi tris paslaugų kokybės lygius, (angl. *Quality of service*) QoS0, QoS1, QoS2 bei neturi jokių integruotų perduodamos informacijos saugos priemonių, todėl perduodamų duomenų saugai užtikrinti tenka naudoti SSL/TLS.
- Analizuojant mokslines publikacijas nepavyko aptikti duomenų apie MQTT protokolo QoS lygių bei duomenų perdavimą naudojant SSL/TLS energijos sąnaudas, todėl priimtas sprendimas atlikti MQTT protokolo saugos ir energijos sąnaudų tyrimą.

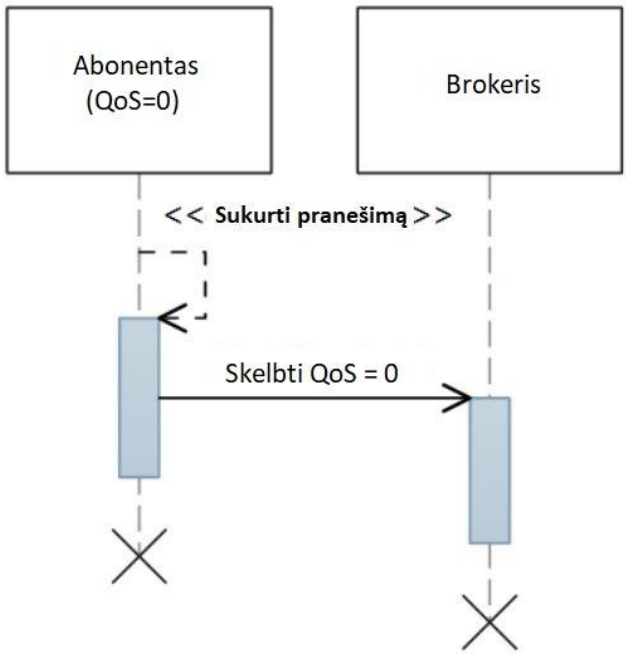
2. Daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistema

Atlikus analizę buvo išnagrinėti daiktų interneto protokolai, apžvelgta, kuo skiriasi jų komunikacija, struktūra, sauga, išsamiai išanalizuotos galimos grėsmės nepasirūpinus tinkamu daiktų interneto komunikacijų saugumu, pasiūlyta specifinių, jau įgyvendintų tiek saugos sprendimų tiek patikimumo sprendimų, taip pat apibrėžta tų sprendimų nauda tolimesniam darbui.

Pagrindinė užduotis, kurią reikia išspręsti šiame darbe – tai padaryti iš nesaugaus MQTT daiktų interneto protokolo jį saugiu, užtikrinti perdavimo patikimumą. Norint užtikrinti saugumą, pasirinktame mikrovaldiklyje bus įgyvendintas komunikavimas su TLS protokolu, užtikrinančiu saugią komunikaciją ir duomenų perdavimą, kad ryšys būtų patikimas bus taip pat naudojamas paslaugų kokybės lygis QoS.

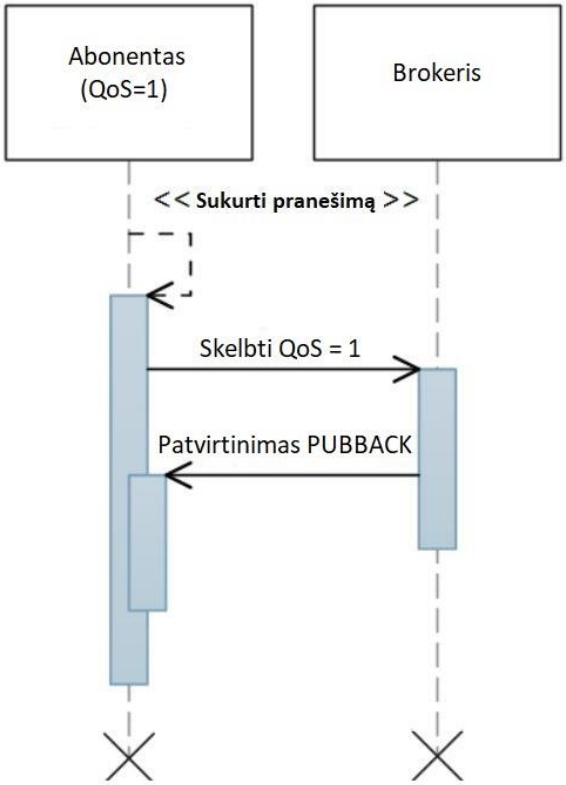
2.1. MQTT protokolo paslaugų kokybės lygiai

Pirmasis paslaugų kokybės lygis dar kitaip vadinamas kaip išsiųsti ir pamiršti (angl. Fire and forget QoS=0) pranešimas pristatomas, bent vieną kartą arba išvis nepristatomas. Sekų diagrama pavaizduota paveiksle 2.1.



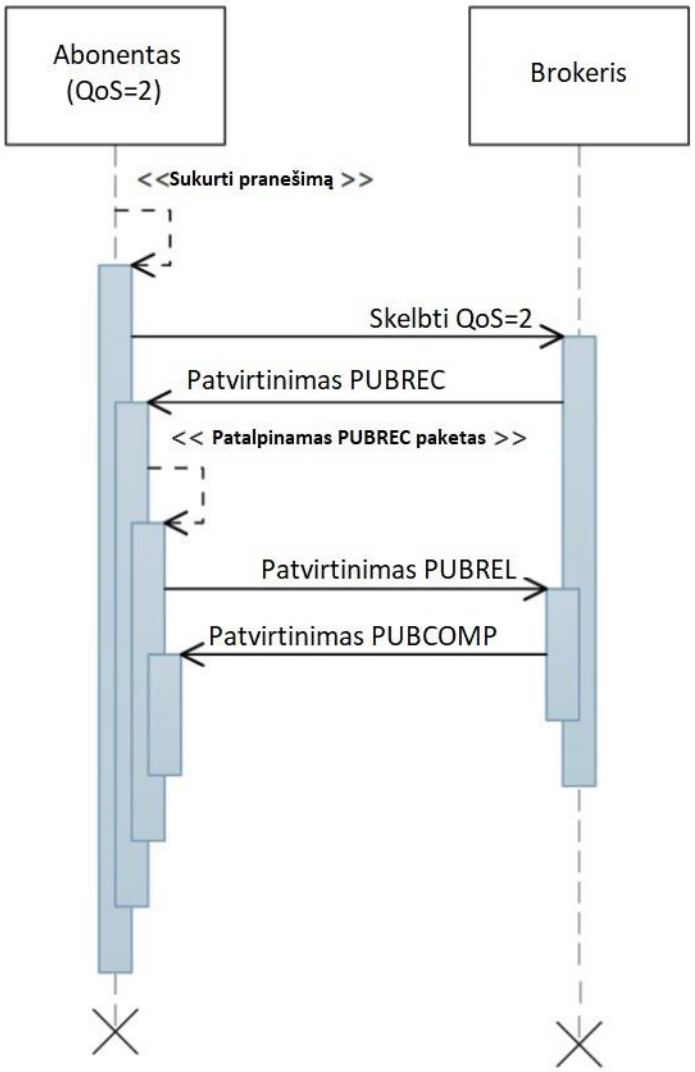
2.1 pav. MQTT paslaugų kokybės lygio QoS = 0 sekų diagrama

Antrasis paslaugų kokybės lygis yra QoS=1. Pranešimas visada pateikiamas bent kartą. Jei siuntėjas negauna patvirtinimo, pranešimas vėl išsiunčiamas su DUP vėliava, kol gaunamas patvirtinimas. Sekų diagrama pavaizduota paveiksle 2.2.



2.2 pav. MQTT paslaugų kokybės lygis QoS = 1 sekų diagrama

Trečiasis paslaugų kokybės lygis (angl. Exactly once QoS=2). Pranešimas visada pristatomas tiksliai vieną kartą. Pranešimas turi būti saugomas vietoje siuntėjo ir gavėjo, kol jis bus apdorotas. Tiksliai vieną kartą yra saugiausias, bet lėčiausias perdavimo būdas. Sekų diagrama pavaizduota paveiksle 2.3.



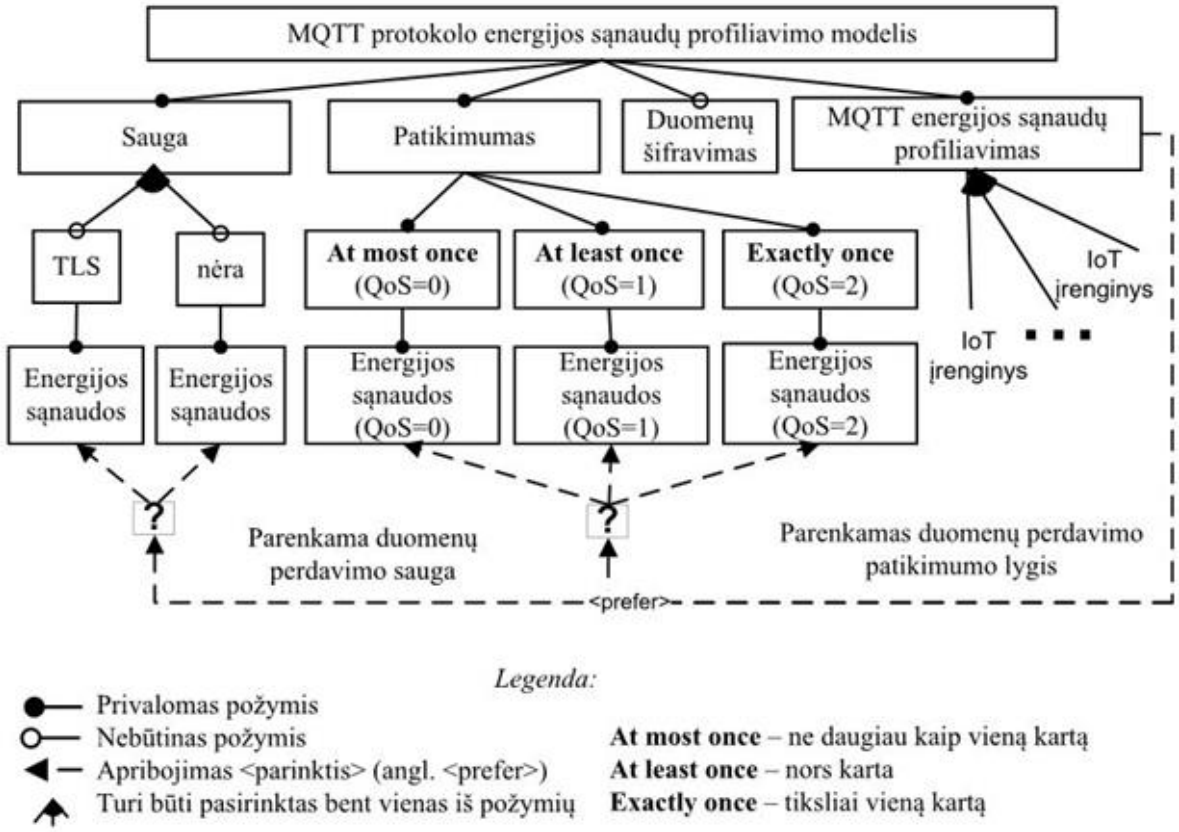
2.3 pav. MQTT paslaugų kokybės lygis QoS =2 sekų diagrama

Panaudojus šiuos paslaugų kokybės lygius yra užtikrinamas pranešimų perdavimo patikimumas.

2.2. Daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistemos modelis

MQTT protokolo profiliavimo sistemos modelis pavaizduotas taikant požymių diagramą (2.4 pav.). Požymių diagrama yra medžio tipo notacija arba nukreiptas ciklinis grafikas, sudarytas iš mazgų rinkinio, ryšių ir apribojimų tarp savybių [23]. Savybė suprantama kaip išorinė matoma elemento savybė (t. y. Sąvoka, objektas, algoritmas, sistema arba domenas). Šaknis yra aukščiausio lygio funkcija. Tarpiniai mazgai sudaro sudėtines savybes ir palieka atstovauti kitas savybes, kurios tam tikroje aplinkoje nėra mažesnės. Briaunos yra naudojamos siekiant laipsniškai suskaidyti junginio funkciją į detalesnes funkcijas. Diagramos kraštai taip pat žymi ryšius arba priklausomybes tarp savybių.

Siūlomame modelyje vertiname patikimumo reikalavimus pagal MQTT paslaugų kokybės lygius su energijos suvartojimu, atsižvelgiant į daiktų interneto sistemų pobūdį, kur energijos vartojimas vaidina svarbų vaidmenį sistemos veikimo metu.

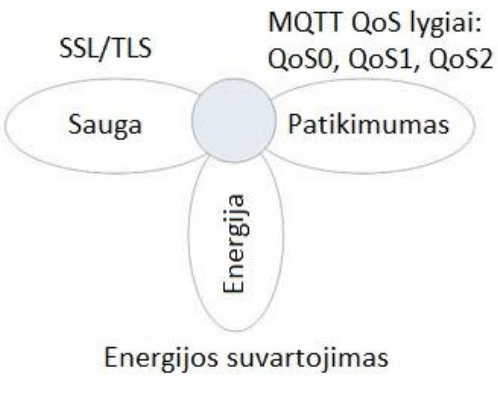


2.4 pav. MQTT protokolo profiliavimo sistemos modelis

Panaudojus šį MQTT protokolo energijos sąnaudų profiliavimo modelį ir pritaikius sistemai, puikiai matosi, kad yra užtikrinama perdavimo sauga, patikimumas parenkant, skirtingus požymius.

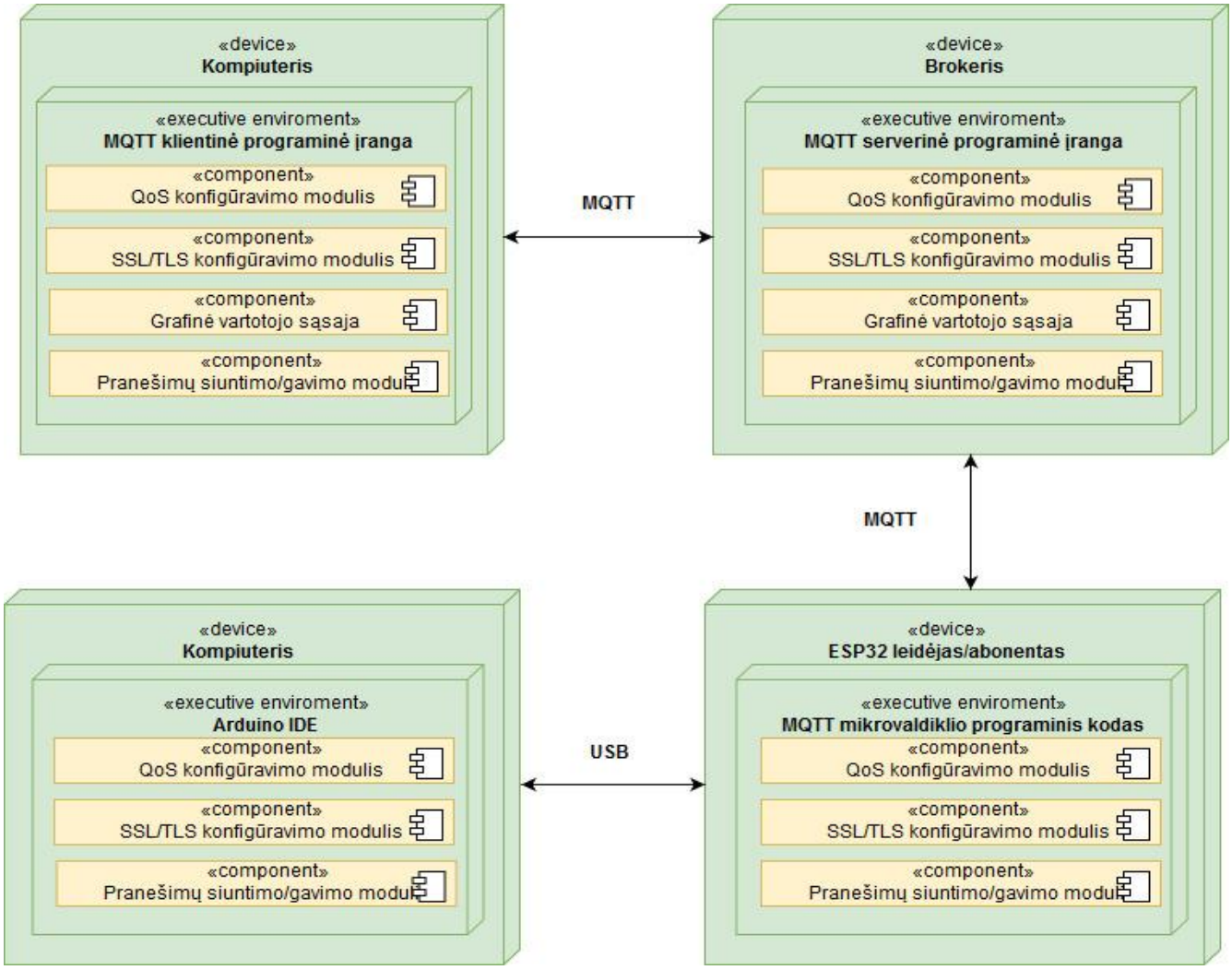
2.3. Daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistemos prototipas

Paveiksle 2.5 pavaizduota magistro darbo sprendimo T formos diagrama kuri susieja tris vertinimo sritis: saugumą, patikimumą ir energijos suvartojimą.



2.5 pav. MQTT protokolo patikimumo, saugos ir energijos sąnaudų susiejimo T formos diagrama

Diagrama taip pat apibūdina pasirinktų domenų kontekstą: saugumo srityje ji yra SSL / TLS, MQTT QoS lygiai užtikrina patikimumą ir energijos suvartojimo matavimą.
Magistro darbe sukurta daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistemos prototipo architektūra, naudojant UML notaciją (pav. 2.6)



2.6 pav. Daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistemos prototipo architektūra

Kompiuteryje esanti sistema su programine įranga yra sudaryta iš keletos elementų (modulių), kurie atlieka tokias funkcijas:

Kompiuterio, brokerio, mikrovaldiklio, programinės įrangos parinktys:

- **Pranešimų siuntimas/gavimas –** abonento užprenumeruotiems pranešimams gauti, arba siųsti.
- **Paslaugų kokybės lygis (QoS) –** skirta pasirinkti paslaugų kokybės lygį patikimumui (QoS0, QoS1, QoS2).
- **SSL/TLS konfigūravimas –** skirtas, srautui šifruoti.
- **GUI (grafinis konfigūravimo langas) –** skirtas atlikti visą konfigūravimą per vartotojo grafinę sąsają.
- **Arduino IDE** – programinė įranga skirta atlikti konfigūracijos valdymą mikrovaldiklyje.

- **MQTT –** pranešimų perdavimo protokolas tarp įrenginių.
- **USB –** skirtas perduoti programinį kodą į mikrovaldiklį.

Atliekant sistemos projektavimą, buvo pasirinkta naudoti, „Mosquitto“ programinę įrangą, kuri būtų naudojama serverinėje pusėje (brokeryje), o kliento pusėje naudota keletas sistemų, tai ta pati „Mosquitto“, bei „MQTTbox“. Pats mikrovaldiklis programuojamas naudojant „Arduino“ platformą vienu atveju buvo naudojamas be srauto šifravimo, kitu atveju buvo panaudotas šifravimo algoritmas (angl. *public key infrastructure* PKI) tarp kliento ir brokerio. Serverinėje dalyje buvo atlikta konfigūracija kuri leidžia varijuoti su skirtingais profiliais, kuomet automatizuotu principu yra parenkamas tam tikras profilis, todėl papildomų nustatymų atlikti daugiau nereikia. „MQTTbox“ konfigūravimas atliekamas naudojant programinės įrangos langą (angl. *graphical user interface* GUI) kuriame galima atlikti įvairius nustatymus, norint perduoti duomenis naudojant MQTT protokolą.

2.7 pav. MQTT klientinės programinės įrangos parinkties konfigūravimas

Naudojant šią programinę įrangą visi nustatymai yra keičiami per grafinę sąsają. Nustatymus galima pasirinkti pagal norimą profilį. Atliekant matavimus tai ganėtinai patogus būdas, atlikti šiuos pakeitimus. Parinkus neteisingus parametrus, pranešimai tiesiog nebus siunčiami, nes visi komponentai esantys šioje grandyje turi būti suderinti taip kaip serverinėje pusėje. Programinės įrangos parinktys:

- **Versija / protokolas –** skirta pasirinkti kurį protokolą ir kurią jo versiją naudoti (MQTT, MQTT su TLS).
- **MQTT kliento vardas –** skirta įrenginio autorizavimui.
- **MQTT kliento ID –** skirta įrenginio autentifikacijai.
- **Abonentas –** pagal IP adresą arba vardą.

Be šių pakeitimų taip pat yra ir kitų, kuriuos taip pat privaloma nustatyti, kad ryšys tarp abonento ir brokerio vyktų be papildomų klaidų ar saugumo spragų, tokių kaip srauto pasiklausymas ar žmogus viduryje atakos arba neteisėtas prisijungimas ir sesijos perėmimas:

- **Vartotojo vardas / slaptažodis –** skirta nustatyti vartotojo vardui ir slaptažodžiui, kad būtų apsauga nuo neautorizuoto vartotojo.
- **Pakartotinio prisijungimo periodas –** skirta pakartotinio pranešimo išsiuntimo periodui.
- **Paskutinės vilties pranešimas –** funkcija kuri suteikia galimybę klientams tinkamai reaguoti į atjungimą.

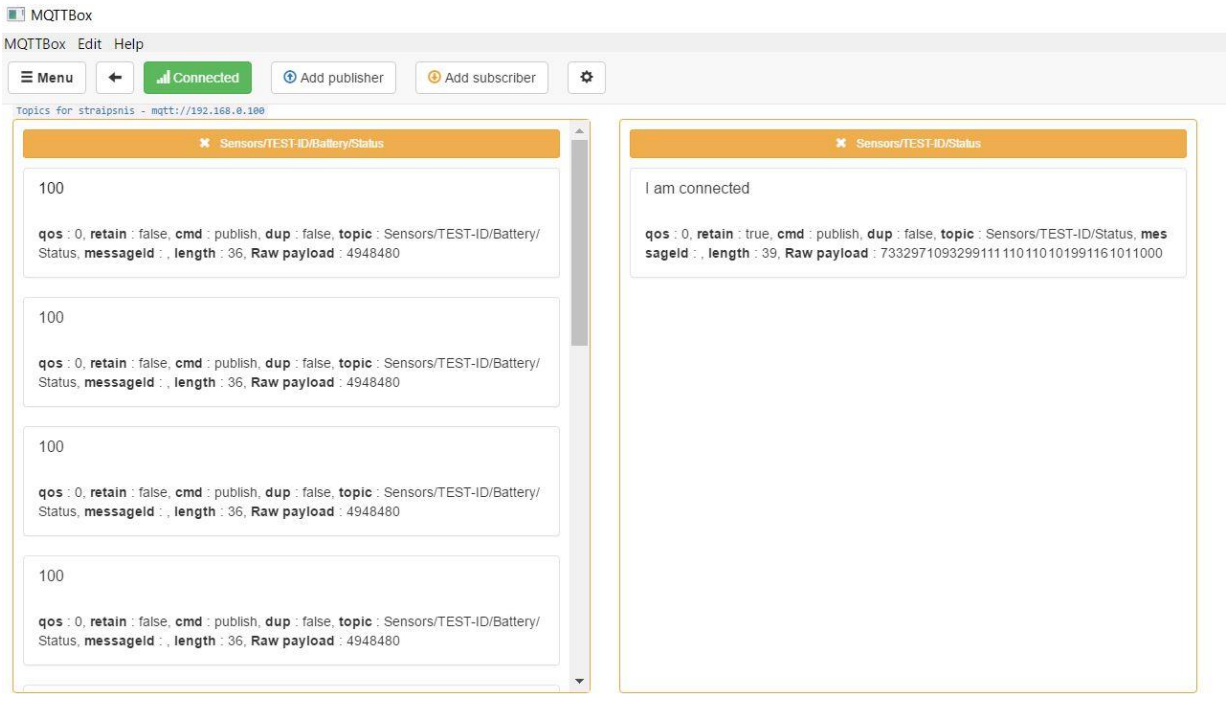
2.8 pav. MQTT klientinės programinės įrangos naudotojo konfigūravimas

Prisijungimui prie abonento yra naudojamas vartotojo vardas ir slaptažodis, taip pat laiko tarpas kas kiek laiko būtų galima persijungti, arba sunaikinti pranešimą jei jis nepristatytas. Toliau galima sukonfigūruoti ir paskutinės vilties pranešimą, tai yra toks momentas, kada siunčiant pranešimą baigiasi energijos sąnaudos, jis pristato paskutinį pranešimą su prierašu „Paskutinės vilties pranešimas“, kad serverinė dalis žinotų, jog pranešimas daugiau nebus siunčiamas, nes abonentas susidūrė tarkime su energijos sąnaudų problema.

Kadangi serverinė dalis yra sukonfigūruota taip, kad priimti, bet kokį sistemos profilį, tai atlikus pakeitimus kliente gali vykdyti pranešimo perdavimą. Naudojant šią sistemą buvo vykdomi šie uždaviniai:

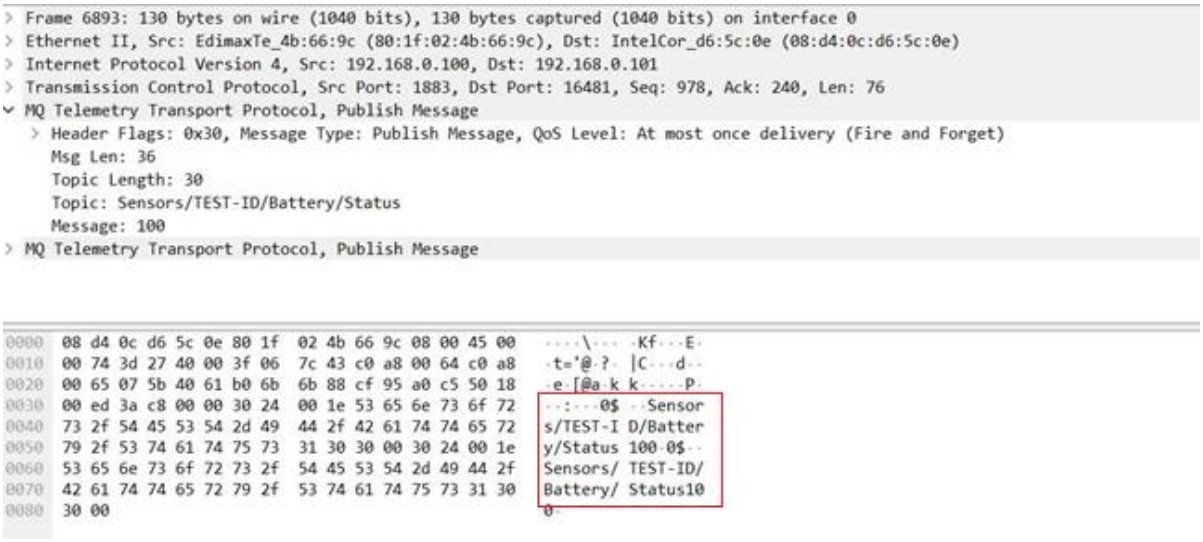
- MQTT protokolo energijos sąnaudų tyrimas
  - MQTT protokolo QoS0 paslaugos kokybės energijos sąnaudų tyrimas
  - MQTT protokolo QoS1 paslaugos kokybės energijos sąnaudų tyrimas
  - MQTT protokolo QoS2 paslaugos kokybės energijos sąnaudų tyrimas

Taip atrodo pranešimų priėmimas naudojant „MQTTbox“ programinę įranga:



2.9 pav. MQTT pranešimų priėmimas

Kadangi šis pranešimo perdavimas buvo atliekamas nenaudojant jokių saugumo priemonių MQTT protokolui, todėl pasileidus programinę įrangą „WireShark“ kuri skirta srauto analizei atlikti, ir atlikus perduoto protokolo analizę galime pamatyti patį pranešimą kuris buvo perduotas. Jis atrodo štai taip:



2.10 pav. MQTT protokolo analizė naudojant „WireShark“ programinę įrangą

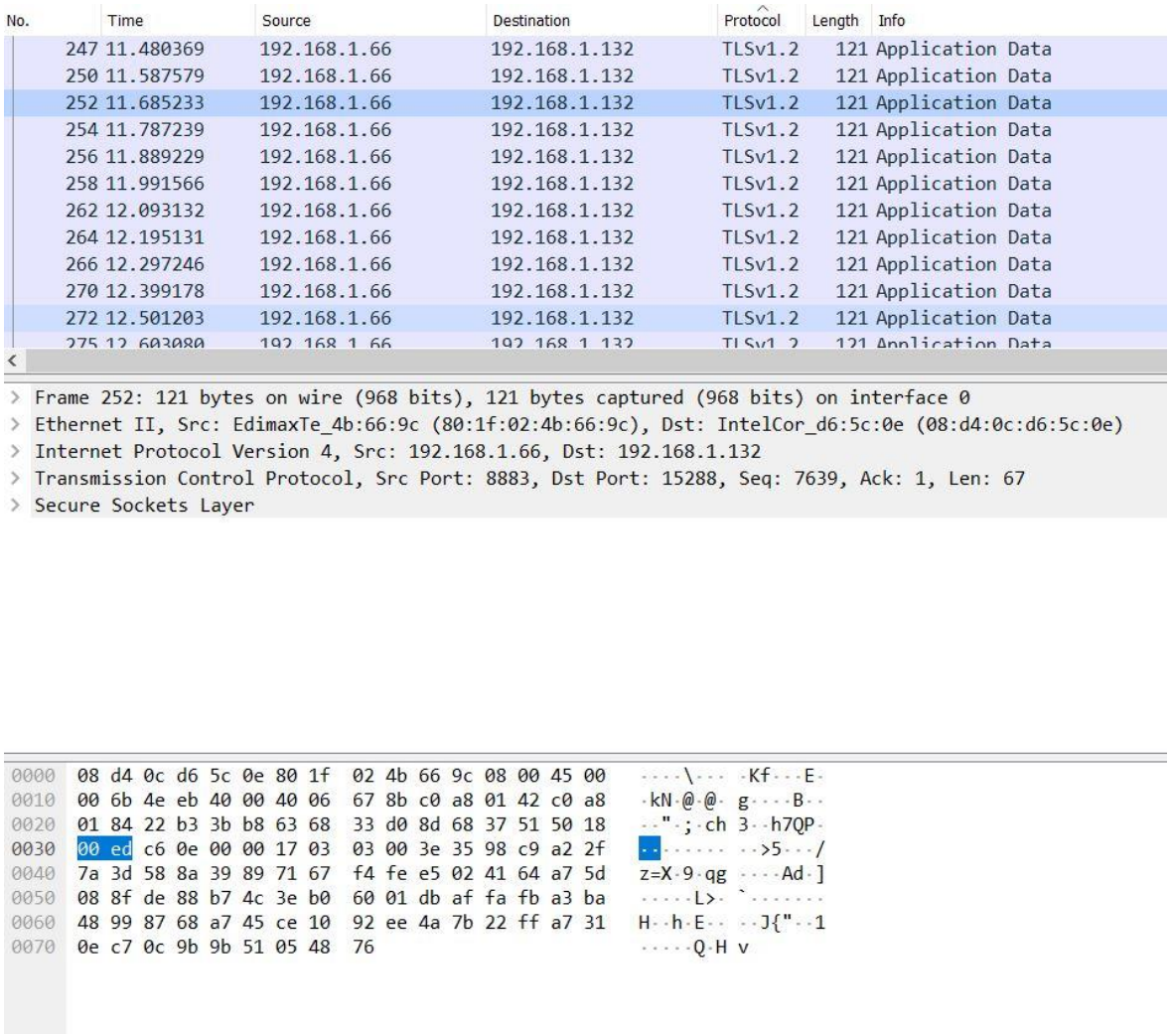
Toks pranešimų perdavimo būdas yra nesaugus ir gali būti naudojamas, tik tokiose sistemose kur nereikalinga užtikrinti duomenų vientisumą, konfidencialumą, autentifikavimą, nes galima perimti visus pranešimus kuriuos perduoda klientas, serveriui ar atvirkščiai.

Kitas sistemos projektavimo ir konfigūravimo sprendimas buvo panaudota ir parinkta „Mosquitto“ programinė įranga ne tik serverinėje dalyje, bet ir kliento dalyje. Konfigūracija keičiama ne taip paprastai kaip naudojant prieš tai minėtą programinę įrangą, reikia išmanyti protokolo struktūrą galimybes, išmanyti kriptografiją, bei turėti daugiau nei bazines žinia atlikti konfigūravimo veiksmus. Atlikus konfigūraciją ir ją išmėginus, galima suskirstyti ir padaryti skirtingus scenarijus, kad nereikėtų konfigūruoti kiekvienu profiliavimo atveju tą patį konfigūravimo tekstą su papildymais. Taip atrodo „Mosquitto“ programinės įrangos konfigūravimo langas:

```
#use_username_as_clientid

# ----------------------------------------------------------------
# Certificate based SSL/TLS support
# ----------------------------------------------------------------
# The following options can be used to enable SSL/TLS support for
# this listener. Note that the recommended port for MQTT over TLS
# is 8883, but this must be set manually.
#
# See also the mosquitto-tls man page.

# At least one of cafile or capath must be defined. They both
# define methods of accessing the PEM encoded Certificate
# Authority certificates that have signed your server certificate
# and that you wish to trust.
# cafile defines the path to a file containing the CA certificates.
# capath defines a directory that will be searched for files
# containing the CA certificates. For capath to work correctly, the
# certificate files must have ".crt" as the file ending and you must run
# "openssl rehash <path to capath>" each time you add/remove a certificate.
cafile C:\Program Files\mosquitto\certs\ca.crt
#capath

# Path to the PEM encoded server certificate.
certfile C:\Program Files\mosquitto\certs\server.crt

# Path to the PEM encoded keyfile.
keyfile C:\Program Files\mosquitto\certs\server.key

# This option defines the version of the TLS protocol to use for this listener.
# The default value allows v1.2, v1.1 and v1.0. The valid values are tlsv1.2
# tlsv1.1 and tlsv1.
tls_version tlsv1.2

# By default a TLS enabled listener will operate in a similar fashion to a
# https enabled web server, in that the server has a certificate signed by a CA
# and the client will verify that it is a trusted certificate. The overall aim
# is encryption of the network traffic. By setting require_certificate to true,
# the client must provide a valid certificate in order for the network
# connection to proceed. This allows access to the broker to be controlled
# outside of the mechanisms provided by MQTT.
#require_certificate false
```

2.11 pav. Programinės įrangos „Mosquitto“ konfigūravimo langas

Naudojant šią programinę įrangą įdiegiau TLS sprendimą MQTT protokolui, kur naudojant šį šifravimą užtikrinamas duomenų perdavimas nuo pradžios iki galo. Skirtingai nei prieš tai naudota sistema kur galima perskaityti pranešimus naudojant „Wireshark“ programinę įrangą, šiuo atveju yra užtikrinamas konfidencialumas, vientisumas, autentiškumas. Kadangi MQTT naudoja užsiprenumeravimo – publikavimo metodą, tad tuo pat metu serveris taip pat gali atlikti ir kliento vaidmenį. Šiuo tiriamuoju atveju MQTT „Mosquitto“ brokeris, skirtas žinučių gavimui bei paskirstymui prenumeratoriams, bus įdiegtas namų kompiuteryje, tad jis atliks brokerio / serverio vaidmenį architektūroje – iš jo bus galima stebėti prisijungiančių klientų skaičių, laiką, siunčiamų žinučių temas. Brokeris atsiunčiamų nešifruotų žinučių klausysis naudodamas 1883 prievadą, o apsaugotų TLS – 8883 prievadą. Taip atrodytų srauto analizė naudojant TLS šifravimą:

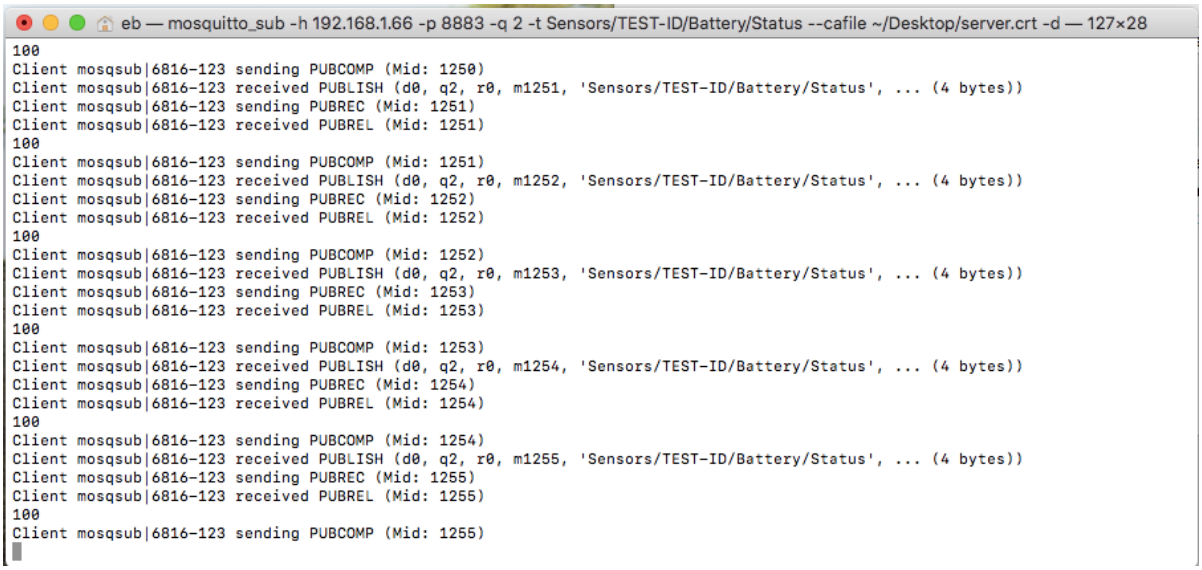


2.12 pav. MQTT su TLS protokolo analizė naudojant „WireShark“ programinę įrangą

Naudojant šią profiliavimo sistemą yra vykdomi šie uždaviniai:

- MQTT protokolo energijos sąnaudų tyrimas naudojant SSL/TLS saugą
  - MQTT protokolo QoS0 paslaugos kokybės energijos sąnaudų tyrimas
  - MQTT protokolo QoS1 paslaugos kokybės energijos sąnaudų tyrimas
  - MQTT protokolo QoS2 paslaugos kokybės energijos sąnaudų tyrimas

Naudojama programinė įranga turi įvykių įrašymo ir stebėjimo galimybę. Todėl perduodamus pranešimus gali stebėti, tinkamai atlikus konfigūraciją. Toliau pateiksiu vieną iš paslaugos kokybės lygių QoS2 pranešimų ir jo pateikimo programiniame lygmenyje. Kadangi naudojant TLS šifravimą iš protokolo analizės negaliu nustatyti ar konfigūracija ir pranešimo persiuntimas yra tikrai tas kurį atlikau. Todėl pasitikrinimui naudojau dviejų faktorių tikrinimą, tai yra „Wireshark“ patikrinti perduodamas pranešimas yra užšifruotas, o ar perduodamas tikrai tas kurį sukonfigūravau paslaugų kokybės lygis panaudojau „Mosquitto“ pranešimo perdavimo monitoringu, kuris atrodo taip:



2.13 pav. Perduodamo pranešimo monitoringas

Iš šios analizės galime matyti, jog pranešimo perdavimas buvo perduotas saugiu TLS šifruotu kanalu ir naudojo QoS2 paslaugų kokybės lygį užtikrinti patikimumą (žr. 2.3 pav.).

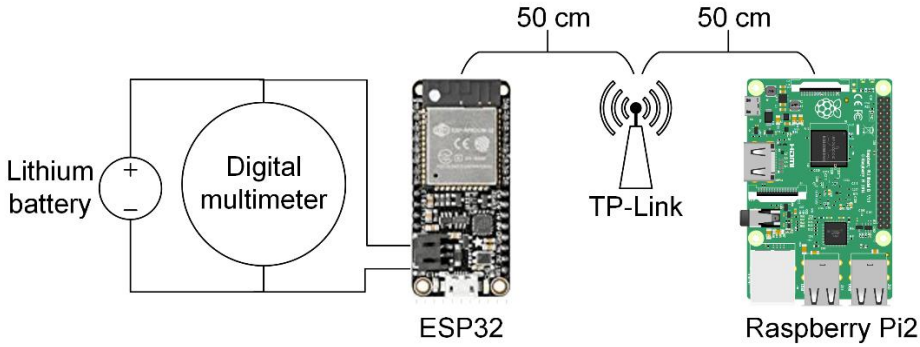
Transporto sluoksnio saugumas ir „Secure Sockets Layer“ (SSL) užtikrina saugų ryšio kanalą tarp kliento ir serverio. Pagrindinėje dalyje TLS ir SSL yra kriptografiniai protokolai, kurie naudoja rankų paspaudimo mechanizmą deryboms dėl įvairių parametrų, kad sukurtų saugų ryšį tarp kliento ir serverio. Pasibaigus rankų paspaudimui, užmegztas šifruotas ryšys tarp kliento ir serverio, ir nė vienas užpuolikas negali stebėti jokios komunikacijos dalies. Serveriai pateikia X509 sertifikatą (kurį paprastai išduoda patikima institucija), kad klientai tikrintų serverio tapatybę. TLS užtikrina, kad trečiosios šalys negalėtų skaityti ar keisti jūsų komunikacijos turinio. MQTT remiasi TCP transportavimo protokolu. Pagal numatytuosius nustatymus TCP jungtys nenaudoja šifruoto ryšio. Norėdami užšifruoti visą MQTT ryšį, daugelis MQTT brokerių leidžia naudoti TLS vietoj paprasto TCP. 8883 prievadas yra standartizuotas saugiam MQTT ryšiui. IANA standartizuotas pavadinimas yra „saugus-mqtt“. 8883 prievadas skirtas tik MQTT per TLS.

2.4. Išvados

1. Pasiūlytas daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistemos modelis.
2. Pasiūlyta daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistemos prototipo architektūra.
3. Sukurtas daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistemos prototipas.

3. Daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistemos eksperimentinis tyrimas

Šiame skyriuje aprašomas sukurtos daiktų interneto įrenginių, naudojančių pranešimų apsikeitimams MQTT protokolą, saugos ir energijos sąnaudų profiliavimo sistemos prototipo eksperimentinis tyrimas ir pateikiami jo rezultatai. Profiliavimo sistemos eksperimento koncepcijos modelis pavaizduotas paveiksle 3.1.



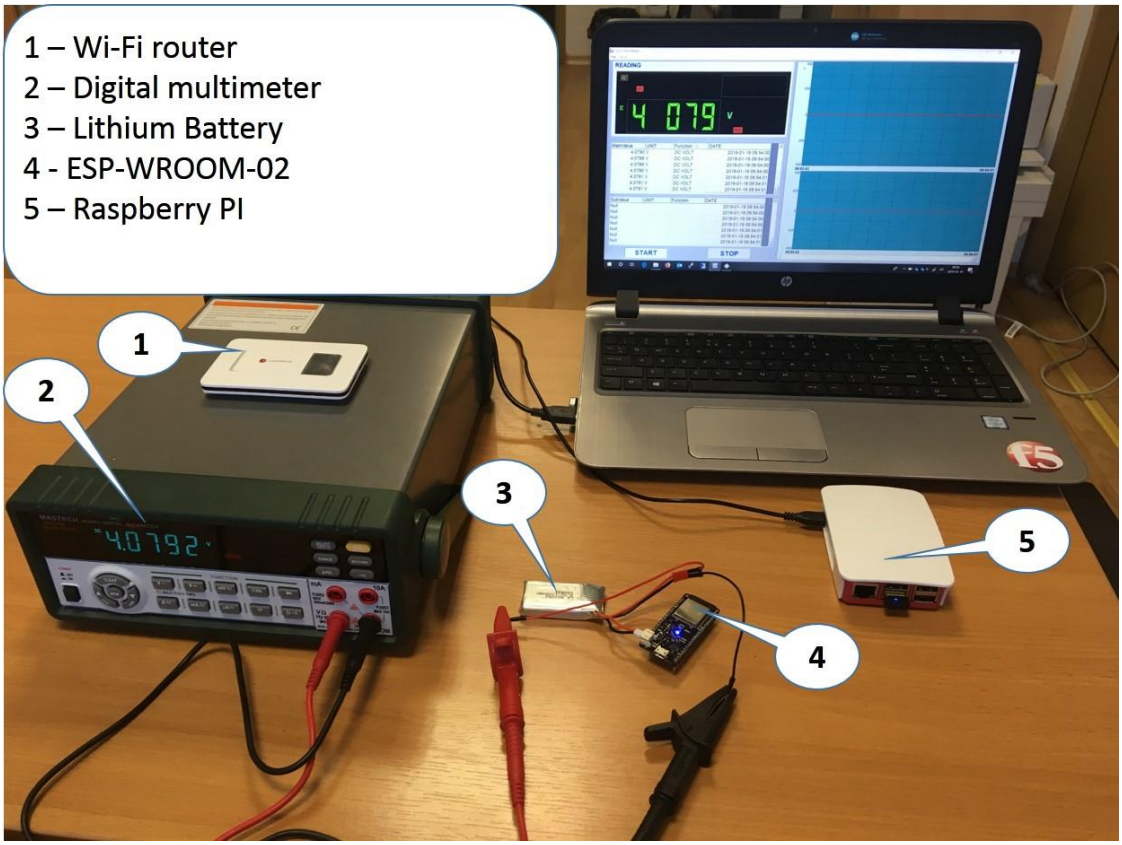
3.1 pav. Sistemos koncepcinis modelis

Profiliavimo sistemos eksperimento koncepcijos modelis sudarytas iš šių komponentų (žr. 3.1 pav.): maršrutizatoriaus, brokerio (RPI2), mikrovaldiklio (ESP32), baterijos, skaitmeninio multimetro.

3.1. Eksperimento aplinka

Naudojantis 2-trame skyriuje aprašytu aprašymu, buvo atliktas eksperimentas. Eksperimento metu buvo naudojama ši programinė įranga:

- Nešiojamasis kompiuteriai „HP probook 450 G2“, „MacBook Air“;
- WiFi maršrutizatorius „Vodafone“ „TP-Link“;
- Mikrovaldiklis „ESP32“ su baterija;
- Brokeris RPI2;
- Skaitmeninis multimetras;



3.2 pav. Eksperimento aplinka

Detali techninės įrangos specifikacija yra pateikiama 3.1 lentelėje.

3.1 lentelė. Tyrime naudos techninės įrangos detali specifikacija

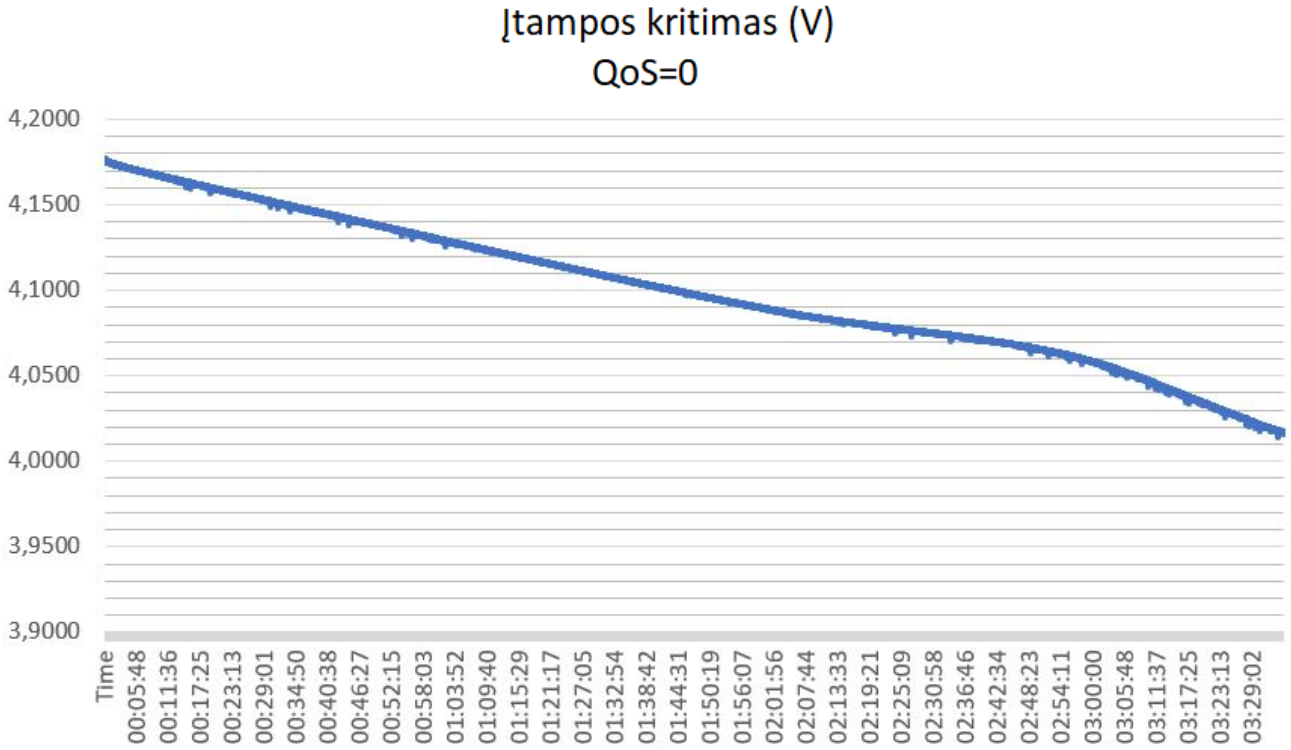
Nešiojamasis kompiuteris „HP probook G2“	
Procesorius	Intel Core i5-6200QM, 2,30 GHz
Operatyvioji atmintis	8 GB
Kietasis diskas	Intel SSD, 250 GB
Grafinė plokštė	Intel HD-520
Tinklo plokštė	Intel Dual-Band AC 3650, a/b/g/n/ac
Operacinė sistema	Windows 10 64-bit
Nešiojamasis kompiuteris „MacBook Air“	
Procesorius	Intel Core i5, 2,80 GHz
Operatyvioji atmintis	8 GB
Kietasis diskas	Apple SSD, 250 GB
Grafinė plokštė	Intel Iris Graphics 6500, 2 GB
Tinklo plokštė	AirPort, a/b/g/n
Operacinė sistema	Mac OS X Yosemite
Mikrovaldiklis su baterija „ESP32“	
Procesorius	Broadcom BCM2837 Arm7 Quad Core 900 MHz
Operatyvioji atmintis	1 GB
Tinklo plokštė	Wi-Fi /b/g/n/
Operacinė sistema	Arduino
Maršrutizatorius ir Wi-fi prieigos stotelė „TP-Link “	
Standartas	Wi-Fi 802.11 b/g/n/ac
Perdavimo greitis	500 Mb/s
Dažnių juosta	2,4 GHz, 5 GHz
Apsaugos protokolas	WPA2

Maršrutizatorius ir Wi-fi prieigos stotelė „Vodafone“	
Standartas	Wi-Fi 802.11 b/g/n/
Perdavimo greitis	200 Mb/s
Dažnių juosta	2,4 GHz, 5 GHz
Apsaugos protokolas	WPA2

Eksperimentui atlikti buvo naudojamas MQTT be šifravimo ir su šifravimu, kurie yra pateikti 2.3 skyriuje.

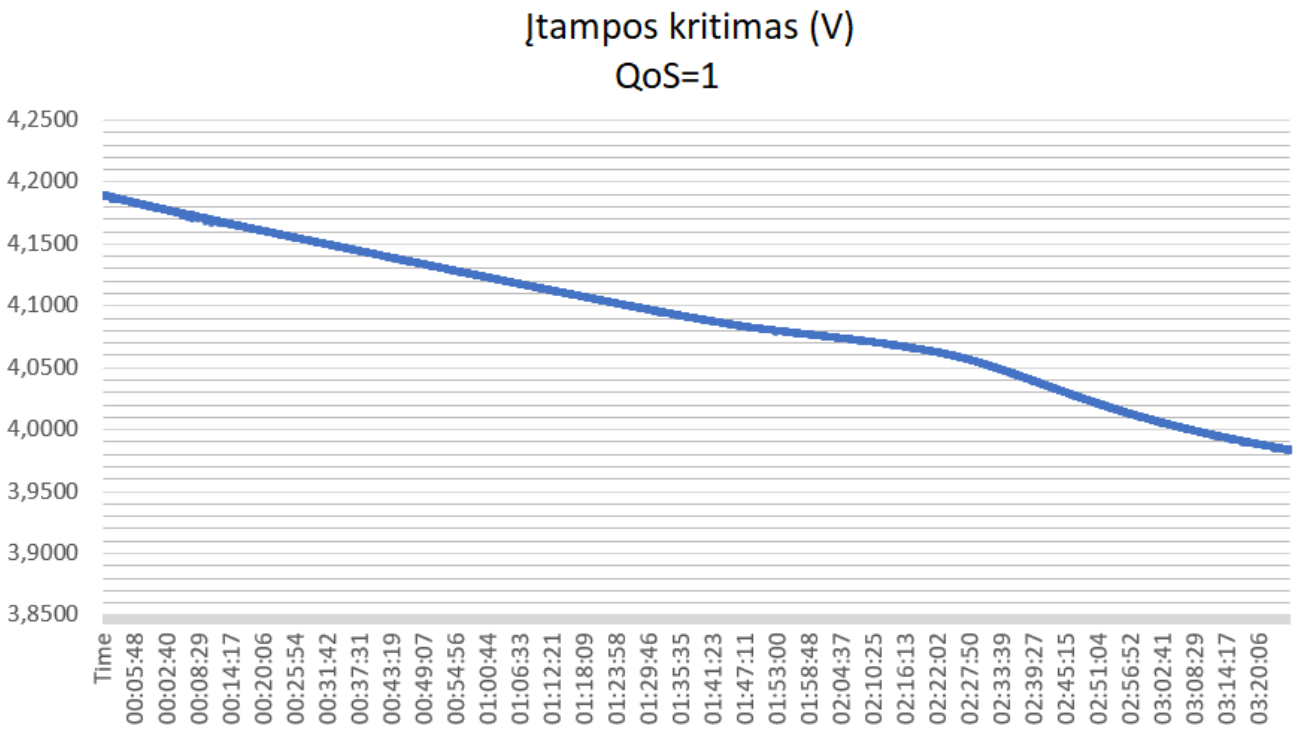
3.2. MQTT paslaugų kokybės lygių energijos sąnaudų tyrimas

Siekiant ištirti MQTT QoS lygių energijos suvartojimą, nustačiau paprastą scenarijų, naudojant eksperimentinę aplinką (žr. 3.2 pav.). Įdarbinau „Raspberry Pi“ ir „ESP-32“ ir sujungiau juos per „Vodafon“ Wi-Fi maršrutizatorių. „Raspberry Pi“ ir „ESP-32“ veikia kaip du tinklo galiniai mazgai - šis paprastas scenarijus gali būti išplėstas, kad tilptų keli ar net keliolika mazgų. Minėti du galiniai mazgai yra nutolę mažiau nei 1 m atstumu vienas nuo kito ir nuo „Vodafon Wi-Fi“ maršrutizatoriaus. Atlikau eksperimentą standartinėse kambario sąlygose. Mūsų eksperimento scenarijaus atveju ESP-32 generuoja duomenų paketus kaip leidėjas. Tuo pačiu metu ESP-32 priima duomenų paketus kaip abonentas. „Raspberry Pi“ atlieka tarpininko vaidmenį, gauna duomenų paketus iš leidėjo ir siunčia gautus duomenų paketus abonentui. Matavimų rezultatai pateikiami (žr. 3.3 – 3.5) pav. parodytas MQTT QoS akumuliatoriaus įtampos lygis „daugiausiai (QoS = 0)“, 5 pav. parodytas MQTT QoS akumuliatoriaus įtampos lygis „Bent kartą (QoS = 1)“ ir 3.5 pav. „MQTT QoS“ įtampos lygis „Tiksliai vieną kartą (QoS = 2)“.



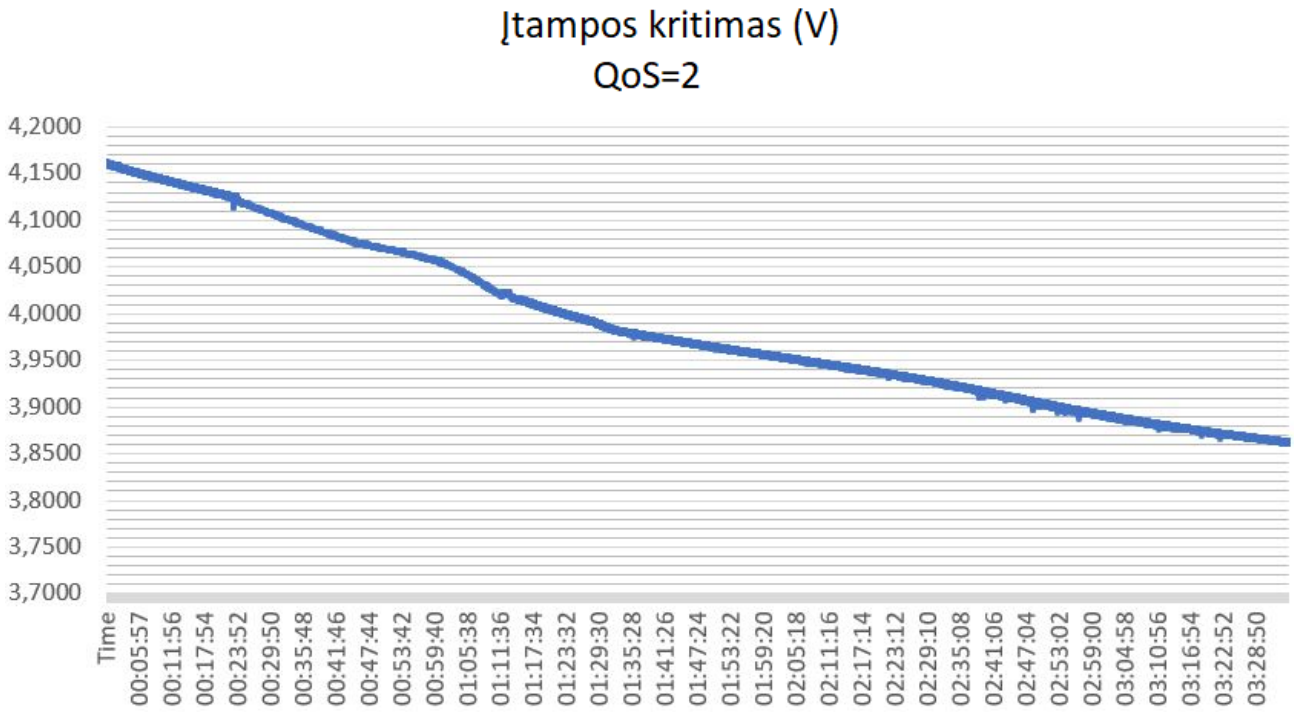
3.3 pav. Įtampos kritimas naudojant paslaugų kokybės lygį 0 (Qos0)

Pav. 3.3 parodo įtampos kritimą per ~3h30min naudojant paslaugų kokybės lygį 0.



3.4 pav. Įtampos kritimas naudojant paslaugų kokybės lygį 1 (QoS1)

Pav. 3.4 parodo įtampos kritimą per ~3h30min naudojant paslaugų kokybės lygį 1.



3.5 pav. Įtampos kritimas naudojant paslaugų kokybės lygį 2 (QoS2)

Pav. 3.5 parodo įtampos kritimą per ~3h30min naudojant paslaugų kokybės lygį 2. Rezultatai apibendrinti 3.2 lentelėje.

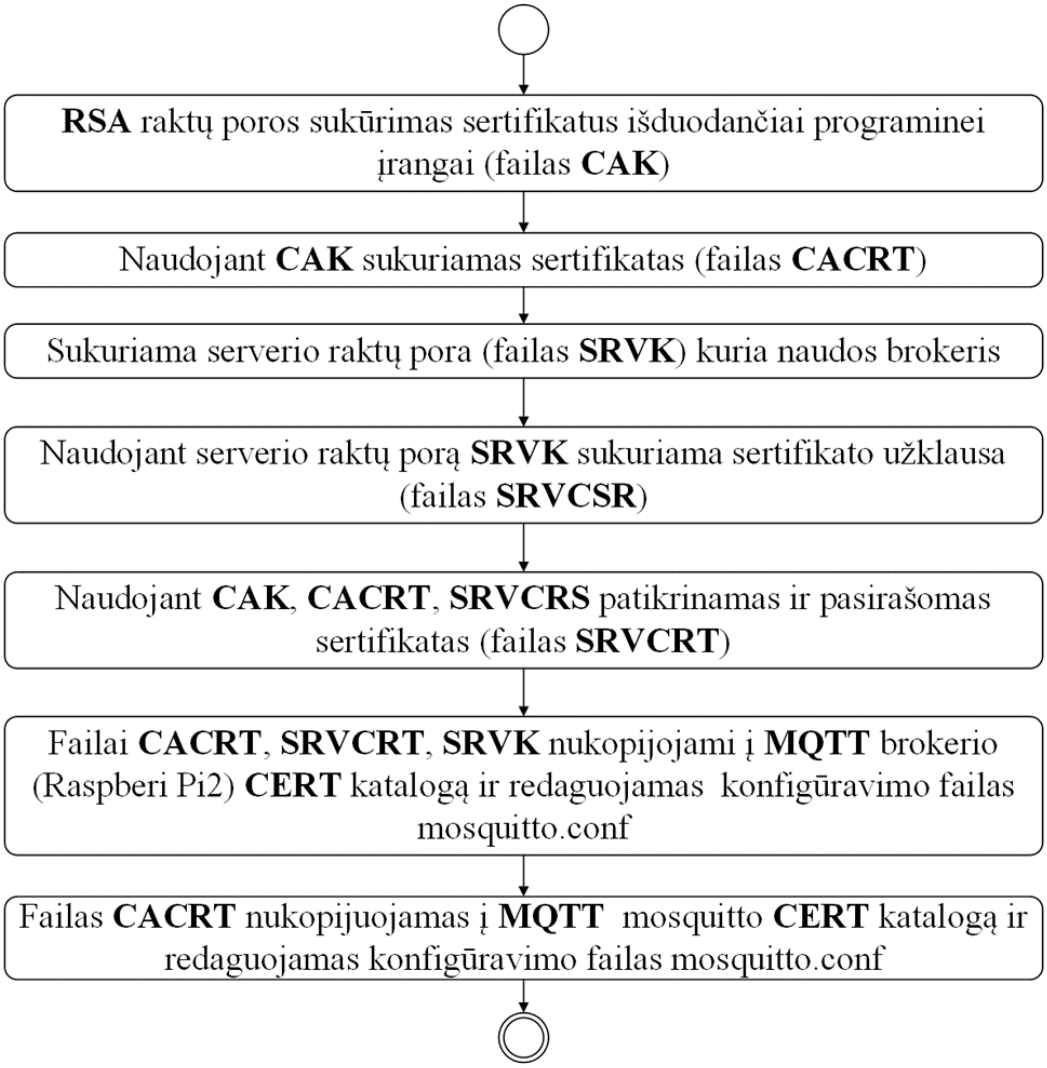
3.2 lentelė MQTT su paslaugų kokybės lygiais rezultatai

MQTT paslaugų kokybės lygiai	Baterijos energijos suvartojimas (V)
QoS=0	0.1596
QoS=1	0.2054
QoS=2	0.2989

Remiantis šiais rezultatais galime įvertinti trijų MQTT protokolo QoS lygių energijos suvartojimo skirtumą.

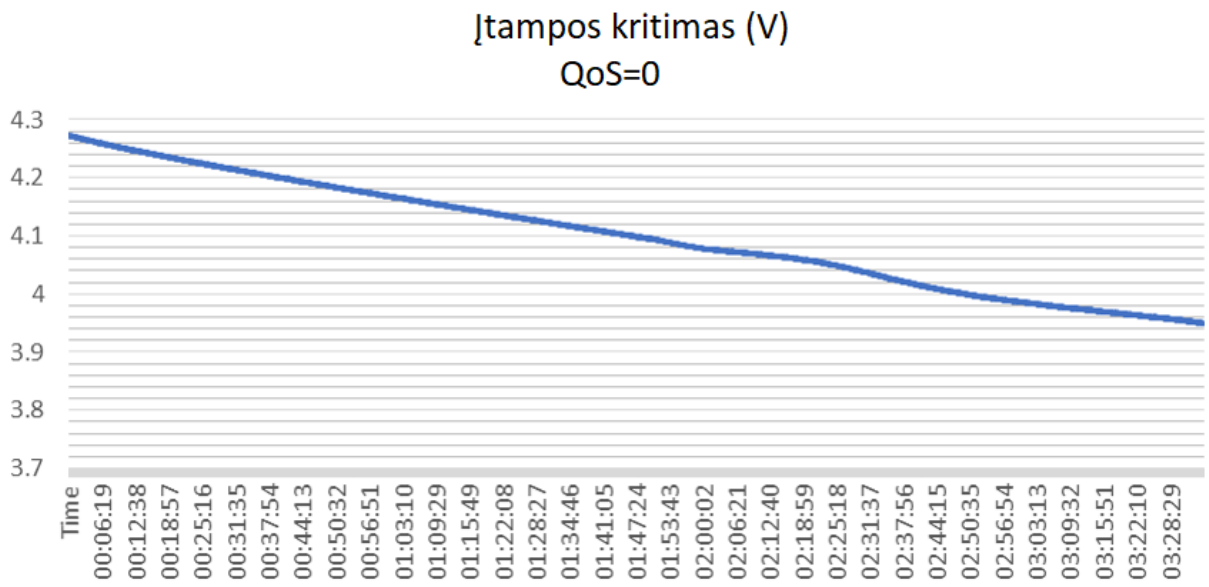
3.3. MQTT paslaugų kokybės lygiai per TLS

ESP32 modulis sujungia ESP8266EX ir yra rekomenduojamas bandymams ar tolesniam tobulinimui. Eksperimentui atlikti naudoju „Mosquitto MQTT“ tarpininką, sukonfigūruotą, kad naudotų TLS. Sukuriu paprastą scenarijų, kad tarpininkas ir klientas sukurtų šifruotą ryšį, panašiai kaip užkoduotas ryšys tarp žiniatinklio serverio ir žiniatinklio kliento. Norėdamas sukurti sertifikatus, naudoju „OpenSSL v1.1.1a“ programinę įrangą „Windows“ operacinėje sistemoje [24]. Mano atveju sukuriu sertifikavimo instituciją (CA) kompiuteryje su „Windows“ operacine sistema. Sertifikato sukūrimas ir įdiegimas „Mosquitto MQTT“ brokeryje (mano atveju „Raspberry Pi2“) ir abonentas / leidėjas (mano atveju ESP32) parodytas pav. 3.6.



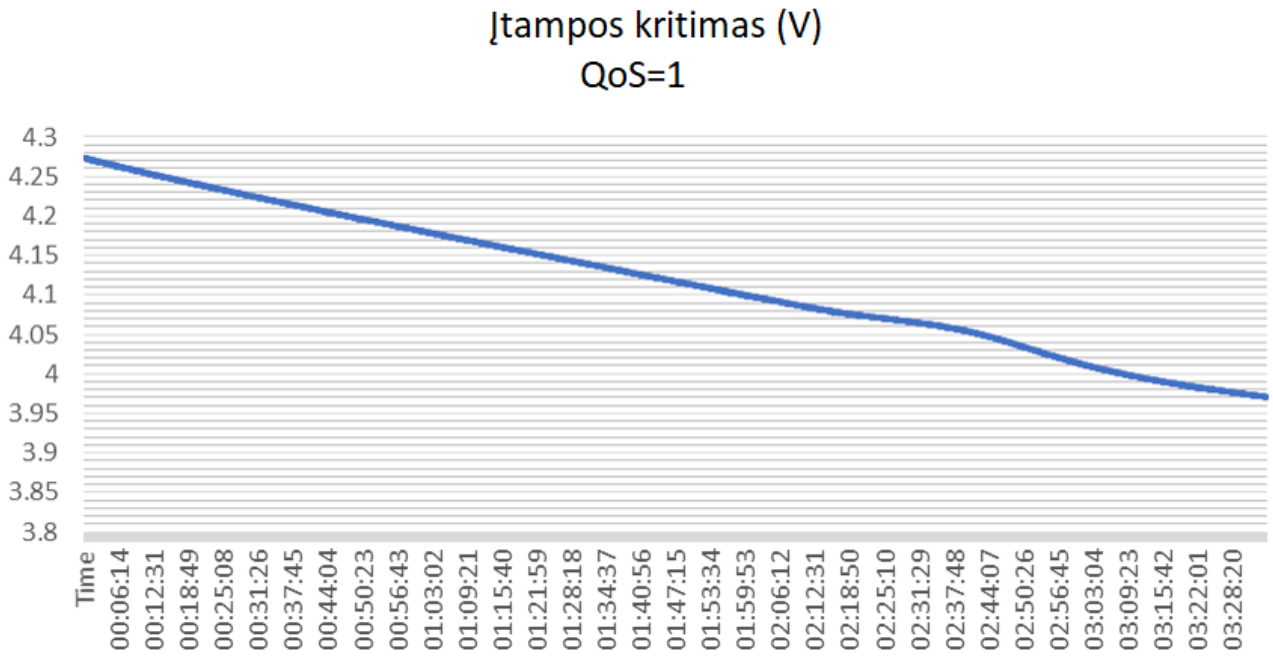
3.6 pav. Sertifikato sukūrimas ir įdiegimas į „Mosquitto“ MQTT brokerį ir prenumeratorių / leidėją

Įdiegus sertifikatą buvo atliekami matavimai su paslaugų kokybės lygiais ir srauto šifravimu. Matavimų rezultatai pateikiami pav. 3.7-3.9. 3.7 pav. parodytas MQTT „At most once“ akumuliatoriaus įtampos lygis naudojant TLS. Įtampos kritimas buvo matuojamas tam tikrą laiko tarpą apie 3val. 30min. su skirtingu profiliu. Iš grafiko matome, kad įtampos kritimas nėra labai didelis įdiegus šifravimą, nors išsiunčiamų pranešimų skaičius su paslaugų kokybės lygiu yra didelis.



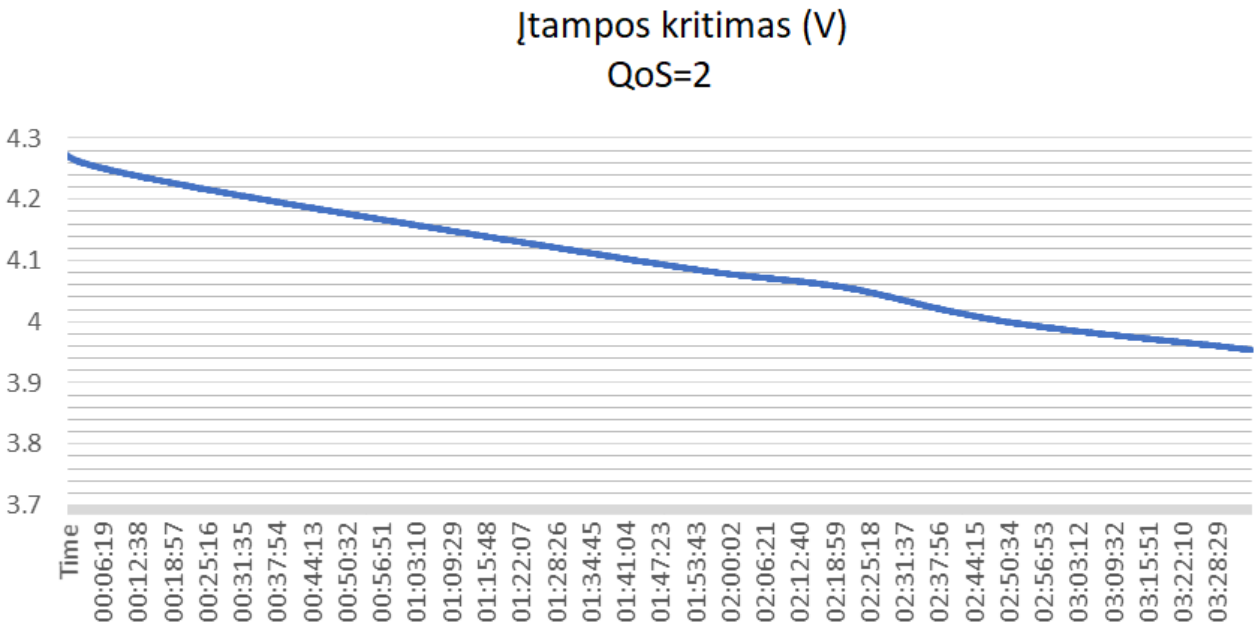
3.7 pav. Įtampos kritimas naudojant TLS ir paslaugų kokybės lygį 0

Pav. 3.8 parodo MQTT “At least once” akumuliatoriaus įtampos lygį naudojant TLS. Šiuo eksperimento metu buvo išmatuotas didžiausias įtampos kritimas.



3.8 pav. Įtampos kritimas naudojant TLS ir paslaugų kokybės lygį 1

Pav. 3.9 parodo MQTT “Exactly once” akumuliatoriaus įtampos lygį naudojant TLS.



3.9 pav. Įtampos kritimas naudojant TLS ir paslaugų kokybės lygį 2

Matavimų rezultatai apibendrinti 3.3 lentelėje. Remiantis šiais rezultatais galime įvertinti trijų MQTT protokolo QoS lygių energijos suvartojimo skirtumą naudojant TLS. Mažiausia energijos sunaudojama QoS1 su TLS - įtampos kritimas 0,3026V. Daugiausia energijos sunaudoja QoS0 su TLS - įtampos kritimas 0,3228V ir QoS2 su TLS sunaudoja daugiau energijos nei QoS1 ir mažiau nei QoS0 - įtampos kritimas 0,3176V.

3.3 lentelė Rezultatų apibendrinimas naudojant paslaugų kokybės lygius su srauto šifravimu

MQTT QoS lygiai	**Energijos suvartojimas**	
	Įtampos kritimas (V)	***Matavimo periodas (hh:mm:ss)***
MQTT “At most once (QoS0)” su TLS	0.3228	03:34:45
MQTT “At least once (QoS1)” su TLS	0.3026	03:34:36
MQTT “Exactly once (QoS2)” su TLS	0.3176	03:34:45

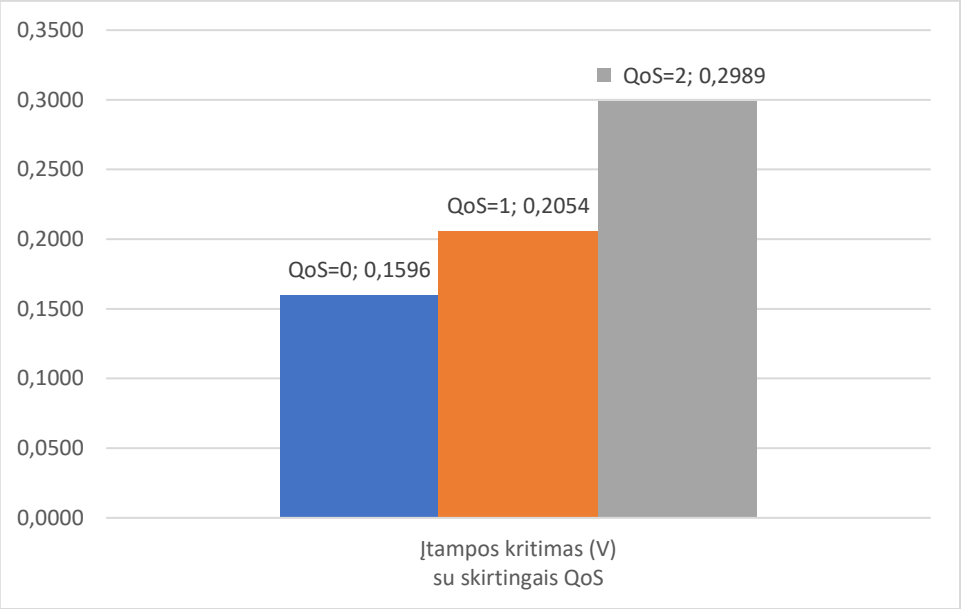
MQTT protokolo energijos sąnaudos, turinčios įvairius QoS su TLS lygius, yra labai skirtingos. Realaus laiko matuojamos MQTT energijos suvartojimo užtikrinimo vertės, palyginti su TLS, pasiekiamos įvairiais QoS lygiais.

3.4. Eksperimento rezultatų apibendrinimas

Šiame skyriuje bus apžvelgti viso eksperimento rezultatai, bei aprašyti ir grafiškai pavaizduoti gauti rezultatai.

3.4 lentelė MQTT eksperimento rezultatų apibendrinimas be TLS ir su skirtingais QoS

	Įtampos kritimas (V) QoS0	**Įtampos kritimas (V) QoS1**	**Įtampos kritimas (V) QoS2**
MQTT	0,1596	0,2054	0,2989

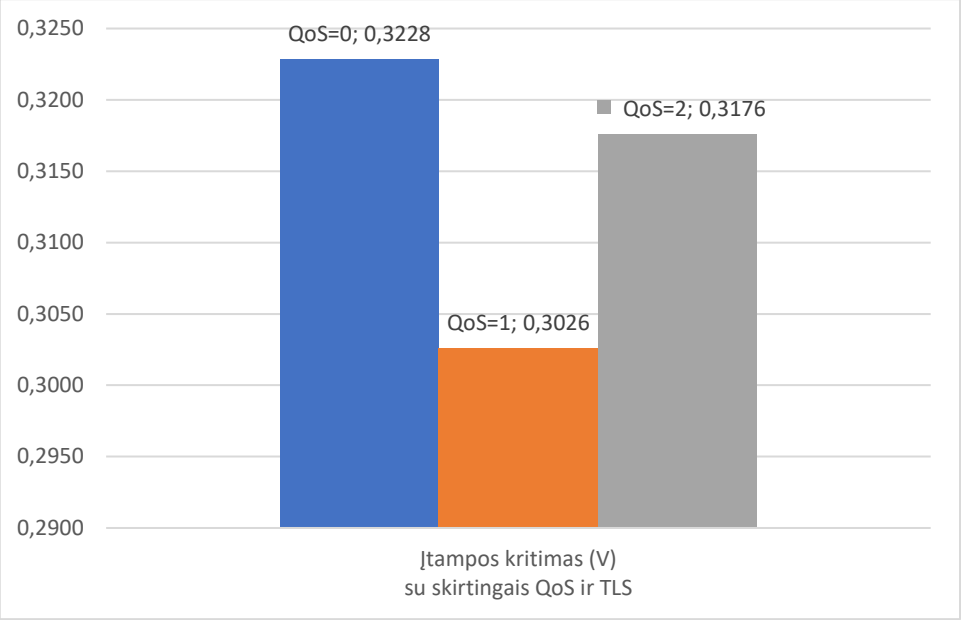


3.10 pav. MQTT protokolo eksperimento rezultatų apibendrinimas be TLS

Iš pav. 3.10 grafiko galima matyti, jog įtampos kritimas yra tendencingas ir didėja naudojant skirtingą paslaugų kokybės lygį. Tarkime su paslaugų kokybės lygiu 0, įtampos kritimas yra mažiausias, o su paslaugų kokybės lygiu 2 yra didžiausias. Paslaugų kokybės lygis 1 yra vidutinis ir optimaliausias variantas, lyginant pranešimo pristatymo pateikiamumą ir patikimumą su įtampos kritimu.

3.5 lentelė MQTT eksperimento rezultatų apibendrinimas su TLS ir skirtingais QoS

	Įtampos kritimas (V) QoS0	**Įtampos kritimas (V) QoS1**	**Įtampos kritimas (V) QoS2**
MQTT su TLS	0,3228	0,3026	0,3176

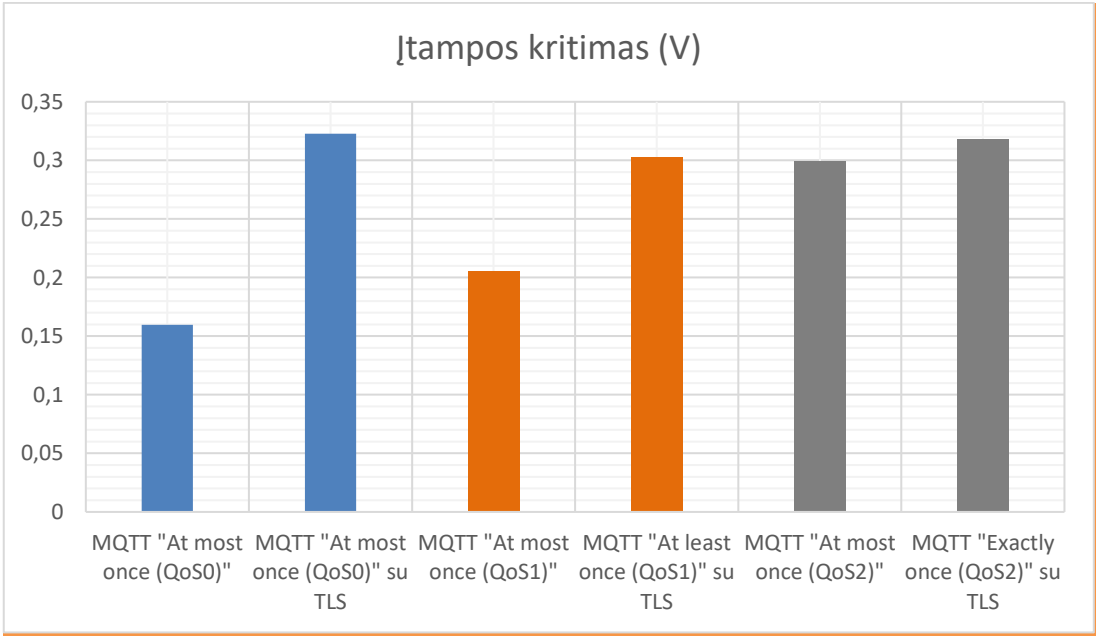


3.11 pav. MQTT protokolo eksperimento rezultatų apibendrinimas su TLS

Įtampos kritimo apibendrinimo rezultatai su TLS pavaizduoti 3.11 pav. tačiau atvirkščiai negu matavimai be srauto šifravimo, rezultatai skirtingi. Nėra tendencingo didėjimo, skirtingais atvejais įtampos kritimas skirtingas.

3.6 lentelė MQTT eksperimento rezultatų apibendrinimas

	Įtampos kritimas (V) QoS0	**Įtampos kritimas (V) QoS1**	**Įtampos kritimas (V) QoS2**
MQTT	0,1596	0,2054	0,2989
MQTT su TLS	0,3228	0,3026	0,3176



3.12 pav. MQTT protokolo eksperimento rezultatų palyginimas

Čia pateikiami apibendrinami rezultatai su skirtingais paslaugų kokybės lygiais ir šifravimu.

3.5. Eksperimentinio tyrimo rezultatų apibendrinimas

1. Surinkta eksperimento aplinka.
2. Atlikta eksperimentinės aplinkos konfigūravimas, taip kad leistų atlikti matavimus.
3. Energijos suvartojimo matavimų rezultatai, atliekant ryšį naudojant MQTT protokolą, gali būti naudojami patikimai prognozuoti trijų QoS lygių energijos suvartojimą:
   a. „QoS“ „Bent kartą (QoS = 1)“ sunaudoja 30% daugiau energijos nei „QoS“ „Ne daugiau kaip vieną kartą (QoS = 0)“;
   b. „QoS“ „Tiksliai vieną kartą (QoS = 2)“ sunaudoja 87% daugiau energijos nei „QoS“ „Ne daugiau kaip vieną kartą (QoS = 0)“;
   c. QoS „Tiksliai vieną kartą (QoS = 2)“ sunaudoja 45% daugiau energijos nei „QoS“ „Bent kartą (QoS = 1);
4. Energijos suvartojimo matavimų rezultatai, atliekant saugų ryšį naudojant MQTT protokolą, gali būti naudojami patikimai prognozuoti trijų QoS lygių energijos suvartojimą:
   a. QoS „Bent kartą (QoS1)” su TLS sunaudoja mažiau energijos nei kiti du QoS lygiai (QoS0 su TLS ir QoS2 su TLS);
   b. QoS TLS daugiau nei vieną kartą (QoS0) sunaudoja daug daugiau energijos nei kiti du QoS lygiai (QoS1 su TLS ir QoS2 su TLS);
   c. QoS Tiksliai vieną kartą (QoS2)“ su TLS sunaudoja 5% daugiau energijos nei QoS „Bent kartą (QoS = 1) su TLS “;
   d. QoS TLS daugiau nei vieną kartą (QoS0) su TLS sunaudoja 6,7% daugiau energijos nei QoS bent kartą (QoS1) su TLS;
   e. QoS „Tiksliai vieną kartą (QoS = 2)” su TLS sunaudoja 1,7% mažiau energijos nei QoS „Daugiausiai (QoS0) su TLS“.

4. Išvados

1. Išanalizuoti daiktų interneto protokolai, nustatyta, kad vienas iš plačiausiai naudojamų protokolų yra MQTT, jį naudoja tokios platformos kaip, gerai žinomos debesų platformos: „Amazon AWS“, „Microsoft Azure“ ir „IBM Watson“.
2. Darbe pasiūlyta MQTT protokolo servisų kokybės lygio ir saugos poveikio energijos suvartojimui įvertinimo ir profiliavimo modelis.
3. Sukurtas profiliavimo sistemos prototipas. Eksperimentinių tyrimų metu atlikti matavimai ir gauti rezultatai rodantys energijos sąnaudas skirtingiems MQTT protokolo servisų kokybės lygiams taikant SSL/TLS saugą.
4. Energijos suvartojimo matavimų rezultatai, atliekant saugų duomenų perdavimą naudojant MQTT protokolą, gali būti naudojami prognozuoti trijų QoS lygių energijos suvartojimą:
    - ✓ QoS „Bent kartą (QoS1)” su TLS sunaudoja mažiau energijos nei kiti du QoS lygiai (QoS0 su TLS ir QoS2 su TLS);
    - ✓ QoS TLS daugiau nei vieną kartą (QoS0) sunaudoja daug daugiau energijos nei kiti du QoS lygiai (QoS1 su TLS ir QoS2 su TLS);
    - ✓ QoS Tiksliai vieną kartą (QoS2)“ su TLS sunaudoja 5% daugiau energijos nei QoS „Bent kartą (QoS = 1) su TLS “;
    - ✓ QoS TLS daugiau nei vieną kartą (QoS0) su TLS sunaudoja 6,7% daugiau energijos nei QoS bent kartą (QoS1) su TLS;
    - ✓ QoS „Tiksliai vieną kartą (QoS = 2)” su TLS sunaudoja 1,7% mažiau energijos nei QoS „Daugiausiai (QoS0) su TLS“.

LITERATŪRA

[1] Gartner Says Worldwide IoT Security Spending Will Reach $1.5 Billion in 2018. STAMFORD, Conn., March 21, 2018. Gartner, Inc. [Online]. Prieiga per internetą: https://www.gartner.com/en/newsroom/press-releases/2018-03-21-gartner-says-worldwide-iot-security-spending-will-reach-1-point-5-billion-in-2018 [Accessed February 19, 2019]

[2] P. N. Howard, "Sketching out the Internet of Things trendline" [Online]. Prieiga per internetą: https://www.brookings.edu/blog/techtank/2015/06/09/sketching-out-the-internet-of-things-trendline/ [Accessed January 25, 2019]

[3] F. Z. Chafi, Y. Fakhri, "The integration of Multi Agent System within the Internet of Things: The use of SigFox shield as a network", SCA '18, October 10–11, 2018, Tetouan, Morocco. [Online]. Prieiga per internetą: https://doi.org/10.1145/3286606.3286835/

[4] B. Safaei, A. M. H. Monazzah, M. B. Bafroei, A. Ejlali, "Reliability Side-Effects in Internet of Things Application Layer Protocols", 2nd International Conference on System Reliability and Safety. 2017 IEEE.

[5] R. Giambona, A. E.C. Redondi, M. Cesana, "Demonstrating MQTT+: An Advanced Broker for Data Filtering, Processing and Aggregation", In 21st ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM '18), October 28-November 2, 2018, Montreal, QC, Canada. ACM, New York, NY, USA. [Online]. Prieiga per internetą: https://doi.org/10.1145/3242102.3243317/

[6] S. B. Kenitar, S. Marouane, A. Mounir, "Evaluation of the MQTT Protocol Latency over Different Gateways", SCA2018, October 2018, Tetuan, Morocco. [Online]. Prieiga per internetą:: https://doi.org/10.1145/3286606.3286864/

[7] M. Singh, R. MA, S. VL, and B. P, "Secure MQTT for Internet of Things (IoT)".2015 Fifth International Conference on Communication Systems and Network Technologies. IEEE Computer Society, 2015, pp. 746-751. DOI 10.1109/CSNT.2015.16

[8] MQTT Version 3.1.1 Plus Errata 01. OASIS Standard Incorporating Approved Errata 01. [Online]. Prieiga per internetą: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/errata01/os/mqtt-v3.1.1-errata01-os-complete.html/ [Accessed February 27, 2019]

[9] Mario Frustaci, Pasquale Pace, Gianluca Aloi, and Giancarlo Fortino, "Evaluating Critical Security Issues of the IoT World: Present and Future Challenges" IEEE internet of things journal, vol. 5, no. 4, rugpjūtis 2018 pp. 2483-2495. DOI 10.1109/JIOT.2017.2767291

[10] W. Ejaz, M. Naeem, A. Shahid, A. Anpalagan, and M. Jo, "Efficient Energy Management for the Internet of Things in Smart Cities". IEEE Communications Magazine, January 2017, pp. 84-91. DOI: 10.1109/MCOM.2017.1600218CM

[11] CoAP (Constrained Application Protocol) over TCP, TLS, and WebSockets, December 18, 2017. Prieiga per internetą: https://tools.ietf.org/id/draft-ietf-core-coap-tcp-tls-11.html#rfc.section.8.1

[12] CoAP Protocol: Step-by-Step Guide November 08, 2018. Prieiga per internetą: https://dzone.com/articles/coap-protocol-step-by-step-guide

[13] F.CHAFI et al 1. Fatima Zahra CHAFI, Youssef FAKHRI. The integration of Multi Agent System within the Internet of Things: The use of SigFox shield as a network. SCA '18, October 10–11, 2018, Tetouan, Morocco. Prieiga per internetą: https://doi.org/10.1145/3286606.3286835

[14] MQTT wiki, Prieiga per internetą: http://mqtt.org/tag/wiki

[15] Soukaina Bakhat Kenitar, Salhaoui Marouane, Arioua Mounir. Evaluation of the MQTT Protocol Latency over Different Gateways. SCA2018, October 2018, Tetuan, Morocco. Prieiga per internetą: https://doi.org/10.1145/3286606.3286864

[16] Khalid Alghamdi, Ali Alqazzaz, Anyi Liu, Hua Ming. 2018. IoTVerif: An Automated Tool to Verify SSL/TLS Certifi cate Validation in Android MQTT Client Applications. In Proceedings of Eighth ACM Conference on Data and Application Security and Privacy, Tempe, AZ, USA, March 19–21, 2018 (CO-DASPY '18), 8 pages. Prieiga per internetą: https://doi.org/10.1145/3176258.3176334

[17] G. Perrone, M. Vecchio, R. Pecori, and R. Giaffreda, "The Day After Mirai: A Survey on MQTT Security Solutions After the Largest Cyber-attack Carried Out through an Army of IoT Devices". In Proceedings of the 2nd International Conference on Internet of Things, Big Data and Security (IoTBDS 2017), p.p. 246-253. DOI: 10.5220/0006287302460253

[18] HiveMQ MQTT company. Prieiga per internetą: https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/

[19] Gautam Srivastava, Andrew Fisher, Robert Bryce, Jorge Crichigno, Department of Mathematics and Computer Science, Brandon University, Brandon, Manitoba, Canada Heartland Software Ardmore, Canada Research Center for Interneural Computing, China Medical University, Taichung, Taiwan, Republic of China College of Engineering and Computing, University of Southern Carolina, Columbia, U.S.A, 2018 November 23, 12 Pages. Prieiga per internetą: https://arxiv.org/pdf/1803.06596.pdf

[20] Bevywise. Natarajan S, Fenzik Joseph D, Ranjith Kumar DSM 2018. Prieiga per internetą: https://www.bevywise.com/mqtt-broker/

[21] Muddsair Sharif, Siegfried Mercelis, Wim Van Den Bergh, Peter Hellinckx 2017 December 20-22. Towards Real-time Smart Road Construction: Efficient Process Management through the Implementation of Internet of Things Prieiga per internetą: https://doi.org/10.1145/3175684.3175721

[22] Peter Membrey, Yuri Demchenko 2015. Cloud Computing Technology and Science (CloudCom), 2015 IEEE 8th International Conference on. IEEE, 2015, pp. 471–476.

[23] V. Štuikys, R. Damaševičius, J. Toldinas, G. Ziberkas, "Matching DSP algorithm transformations for power, performance and memory trade-offs", in Information Technologies' 2009: proceedings of the 15th International Conference on Information and Software Technologies, IT 2009, Kaunas, Lithuania, April 23-24, 2009, Kaunas University of Technology. Kaunas: Technologija. pp. 178-186.

[24] 20-Nov-2018 OpenSSL 1.1.0j is now available, including bug and security fixes [Online]. Available: https://www.openssl.org [Accessed January 4, 2019]

[25] Marko Pavelic, Vatroslav Bajt, Mario Kusek. Energy efficiency of Machine-to-Machine protocols 2018 May 21-25 pp. 361-366.

[26] J. L. Espinosa-Aranda, N. Vallez, C. Sanchez-Bueno, D. Aguado-Araujo, G. Bueno, O. Deniz, "Pulga, a tiny open-source MQTT broker for flexible and secure IoT deployments". 1st IEEE Workshop on Security and Privacy in the Cloud, Florence (Italy), September 30, 2015

[27] Jevgenijus Toldinas, Borisas Lozinskis, Edgaras Baranauskas. „Evaluation of the impact on energy consumtion of MQTT protocol over TLS" Prieiga per internetą: http://ivus.vdu.lt/wp-content/uploads/2019/04/IVUS-2019-conference-proceedings.pdf pp. 72-76 April 2019.

7. PRIEDAI

1. Straipsnis;
2. Geriausio pranešimo apdovanojimas;

7.1. Straipsnis

Evaluation of the impact on energy consumption of MQTT protocol over TLS

Edgaras Baranauskas
Department of Computer Sciences
Kaunas University of Technology
Kaunas, Lithuania
edgaras.baranauskas@ktu.edu

Jevgenijus Toldinas
Department of Computer Sciences
Kaunas University of Technology
Kaunas, Lithuania
eugenijus.toldinas@ktu.lt

Borisas Lozinskis
Department of Computer Sciences
Kaunas University of Technology
Kaunas, Lithuania
borisas.lozinskis@ktu.lt

***Abstract*— Message Queuing Telemetry Transport (MQTT) protocol is widely used in device-to-device communications. While MQTT has three quality of service (QoS) levels, it does not integrate security mechanisms. Transport Layer Security (TLS) is the standard protocol on top of the Transmission Control Protocol (TCP) to secure data in communications. In this paper, we evaluate the impact on energy consumption of MQTT protocol using its QoS levels over TLS.**

Keywords—IoT, MQTT, TLS, battery energy consumption

I. Introduction

According to Gartner prediction spending on Internet of Things (IoT) endpoint security solutions worldwide will reach 631 millions of dollars in 2021 [1]. The term Internet of Things define smart objects that are interconnected using various network interfaces and protocols such as Constrained Application Protocol (CoAP), Message Queuing Telemetry Transport (MQTT), MQTT-SN (for sensor networks), Extensible Messaging and Presence Protocol (XMPP), Web Application Messaging Protocol (WAMP) and many others. In machine-to-machine (M2M) application layer protocols most popular is MQTT, well-known cloud platforms, such as Amazon AWS, Microsoft Azure, and IBM Watson expose their services through MQTT [2]. MQTT has a low memory footprint, low power consumption, and better distribution of information to recipients [3]. Because of that, MQTT protocol is widely used in device-to-device (D2D) communications, where one of the major issue is to ensure the security of devices and D2D communications [4]. MQTT has three quality of service (QoS) levels and does not integrate security mechanisms. Transport Layer Security (TLS) is the standard protocol on top of the Transmission Control Protocol (TCP) to secure data in communications. OASIS [5] explicitly recommends the utilization of TLS as security decision at transport layer. In this paper, we evaluate the impact on energy consumption of MQTT protocol using its QoS levels over TLS.

II. Related work

In [4] authors declare the user's responsibility to address security issues for MQTT, MQTT-SN protocols and suggests enabling security for them by envisaging SSL/TLS, but due to IoT heterogeneity it is cumbersome to manage certificates and keys. Thus, authors [4] propose attribute based encryption for secure MQTT that augments security feature for the existing MQTT protocol and its variants.

Use of Datagram Transport Layer Security (DTLS) for securing data communications over User Datagram Protocol (UDP) adds at least 33 bytes to the original packet header, and while IoT devices run on batteries, efficient secure communication scheme is needed [6].

A novel security mechanism introduced for MQTT environments is based on AugPAKE via a secure side channel, where authentication and authorization tokens are transported in the same field [7] of the topic name. In [8] the most known application layer protocols are compared: CoAP, MQTT, XMPP, HTTP, AMQP and WebSocket, All the protocols mentioned above use TCP as transport layer (CoAP uses UDP) and TLS/SSL as security layer (CoAP uses DTLS). In terms of Message Oriented Approach (MOA), MQTT stands out [8]. Requirements for authentication, authorization, data integrity, and confidentiality do not included in the MQTT specification. Authors [9] argue that the lack of security requirements in the MQTT protocol standard is related to:

- MQTT focuses only on message dispatching.
- Reducing the overhead that is related to security features is used to keep the protocol as light as possible.
- Historical implementations of MQTT were based on private networks.
- Significantly different security functionalities required while MQTT is used from IoT devices to Facebook messenger mobile application.

The authors [9] are inclined to believe that a good mid-term solution to large-scale MQTT security problems could be represented by implementation of TLS. Current open-source MQTT implementations compared in table I.

TABLE I. Open-source MQTT implementation

MQTT implementation	MQTT implementation property		
	Definition	*Security*	*QoS*
Mosquitto [10]	Most commonly used implementation	SSL/TLS support	QoS0, QoS1, QoS2
eMQTTC [11]	Asynchronous Erlang MQTT Client Requires Erlang R17+	TCP/SSL Socket Support	QoS0, QoS1, QoS2
Apollo [12]	Is a faster, more reliable, easier to maintain messaging broker built from the foundations of the original ActiveMQ	SSL/TLS Support	QoS0, QoS1, QoS2
Artemis [13]	Implementation arising from ActiveMQ	SSL support	QoS0, QoS1, QoS2

In [14] proposed potential methodologies to extend the Common Architectures and Network services found in the IEEE 1451 Family of Standard into applications that utilize MQTT. The authors installed the Mosquitto MQTT client

onto ESP-32s, MQTT broker onto Raspberry Pi 3 and experimentally conclude that MQTT is an effective communication protocol when it comes to small-scale systems, security is a major area for future investigation. MQTT has its downsides in security but is being greatly adapted in the world of IoT today and the hope is extend that adaptation to the IEEE 1451 Family of standards [14].

MQTT is a simple protocol designed for devices with low processing power and it tries to minimize the processing needed to exchange messages, which means that serious security problems arise such as lack of: authentication, authorization, confidentiality and integrity [15].

The security challenges of the IoT industry with focus on standardized communication protocols explored and implementation details for the security levels mandated by the Constrained Application Protocol provided in [16]. MQTT implementations also offer out of the box the security certificates mode that could be achieved in the Java Paho library or as part of the Mosquitto framework. In fact, the MQTT broker also offers the possibility to maintain a list of revoked certificates that can be used to disable rogue endpoints [16].

The most critical issues with the aim of guiding future research directions on the IoT security panorama highlighted [17]. According to the author conclusion, the most vulnerable level of the IoT system model is the perception layer due to the physical exposure of IoT devices, to their constrained resources and to their technological heterogeneity. Thus, it is crucial, in the next future, to start working on the critical issues of this level implementing lightweight security solutions that can adapt to the heterogeneous environments with resource-constrained devices.

Smart city solutions have to be energy-efficient, cost-efficient, reliable, secure, to do that IoT devices should operate in a self-sufficient way without compromising QoS in order to enhance the performance with uninterrupted network operations. Therefore, the energy efficiency and life span of IoT devices are key to next generation smart city solutions [18]. With the increase in IoT applications for smart cities, energy-efficient solutions are also evolving for low-power devices. Energy-efficient solutions such as Lightweight Protocols, Scheduling Optimization, and Predictive Models for Energy Consumption, Cloud-Based Approach, Low-Power Transceivers, and Cognitive Management Framework can reduce energy consumption or optimize resource utilization. Possible future directions for energy management in smart cities are [18]:

- Energy-efficient mechanisms for software-defined IoT solutions, which can provide scalable and context-aware data and services.
- Directional energy transmission from dedicated energy sources for wireless power transfer.
- Energy efficiency and complexity of security protocols are crucial aspects for their practical implementation in IoT; thus, it is important to investigate robust security protocols for energy constraint IoT devices.
- Fog computing can lead to energy saving for most of the IoT applications; therefore, it is important to study energy consumption of fog devices for IoT applications.

In [19] authors evaluate MQTT (QoS0) vs HTTPS, send performance, battery energy consumption and conclude that while HTTPS is slightly more efficient in terms of establishing connection, MQTT is much more efficient during transmission.

III. MQTT QUALITY OF SERVICE LEVELS

MQTT provides three levels of QoS [20]:

At most once (Fig. 1) - sometimes called "fire and forget". The message is delivered at most once, or it is not delivered at all.

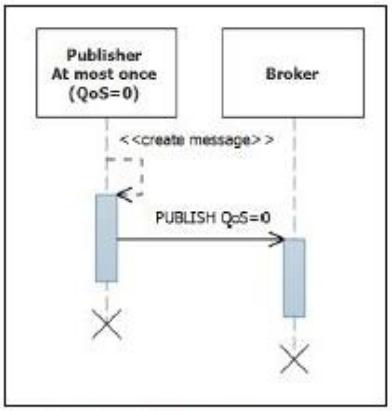


Fig. 1. MQTT QoS level "At most once"

At least once (Fig. 2), it is the default mode of transfer. The message is always delivered at least once. If the sender does not receive an acknowledgment, the message is sent again with the DUP flag set until an acknowledgment is received.

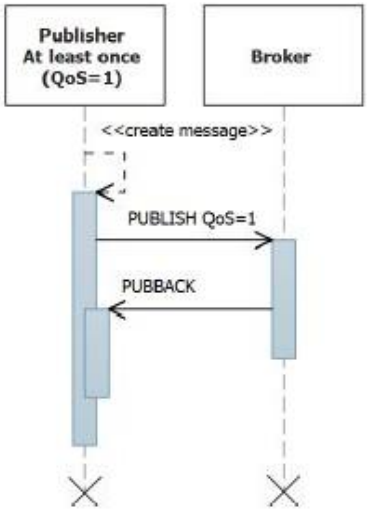


Fig. 2. MQTT QoS level "At least once"

Exactly once (Fig. 3), the message is always delivered exactly once. The message must be stored locally at the sender and the receiver until it is processed. **Exactly once** is the safest, but slowest mode of transfer.

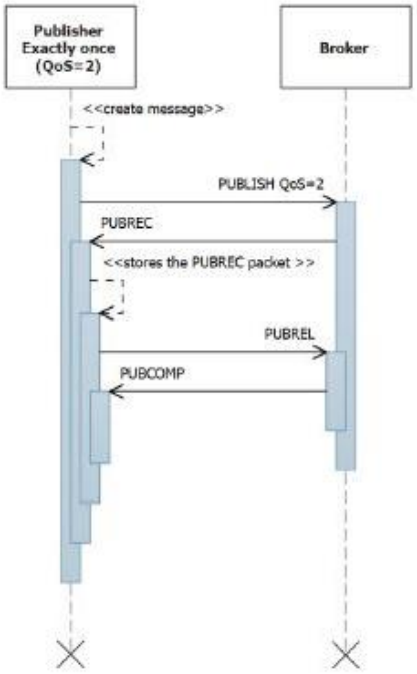


Fig. 3. MQTT QoS level "Exactly once "

IV. Evaluation framework and experimental setup

A general framework for evaluation of the impact on energy consumption of MQTT protocol over TLS is shown in Fig. 4. T-diagram is linking together three evaluation domains: security, reliability and energy consumption.

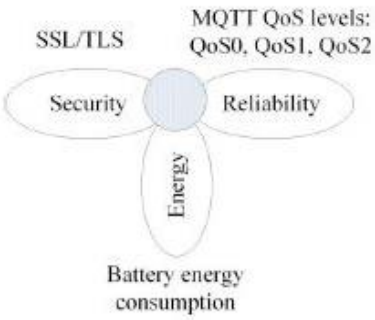


Fig. 4. General framework for evaluation of the impact on energy consumption of MQTT protocol over TLS

The framework also outlines a context of the selected domains: for security domain it is SSL/TLS , the MQTT QoS levels ensure reliability, and battery energy consumption measurement for energy domain. Our experiments are performed using (see Fig. 5):

- **Access point** – Wi-Fi router TP-Link.
- **Broker** – Raspberry Pi2 with Broadcom BCM2837 Arm7 Quad Core CPU, clock frequency 900MHz, 1GB RAM, 802.11b/g/n Wi-Fi communication protocols.
- **Subscriber/Publisher** – IoT Module ESP32 with Tensilica L106, 32-bit, RISC CPU, clock frequency 160 MHz, 802.11b/g/n Wi-Fi communication protocols.
- **Measuring instrument** – digital multimeter MASTECH MS8050.
- **Power supply** for ESP32 – lithium battery LS903052, 3.7V, 1200mAh.

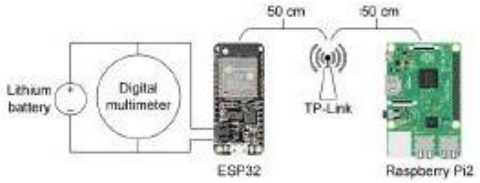


Fig. 5. Experimental setup

The ESP32 module integrates ESP8266EX is and is recommended for tests or for further development. For our evaluation, we use the Mosquitto MQTT broker that configured to use TLS. We create a simple scenario to establish encrypted connection between broker and client similarly as encrypted connection between web server and web client. To create certificates we use OpenSSL v1.1.1a software for Windows [21]. In our case, we create Certification authority (CA) in a computer with Windows OS. Certificate creation and installation in the Mosquitto MQTT broker (in our case Raspberry Pi2) and in the subscriber/publisher (in our case ESP32) is shown in Fig.6.

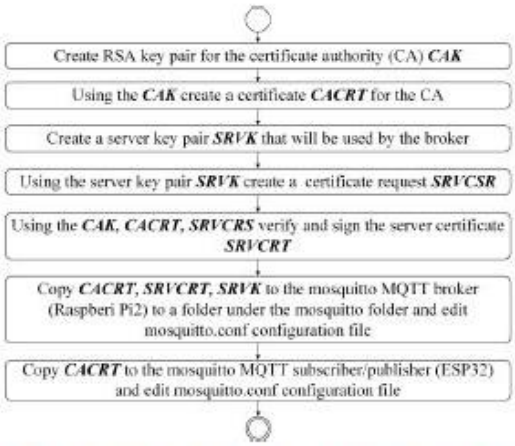


Fig. 6. Certificate creation and installation in the mosquitto MQTT broker and subscriber/publisher

V. Experimental results

The results of measurements are presented in Figs. 7-9. Fig. 7 shows the battery voltage level for MQTT "At most once" over TLS,

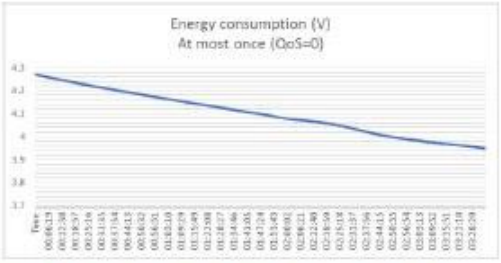


Fig. 7. Battery voltage level for MQTT "At most once" over TLS

Fig. 8 shows the battery voltage level for MQTT "At least once" over TLS.

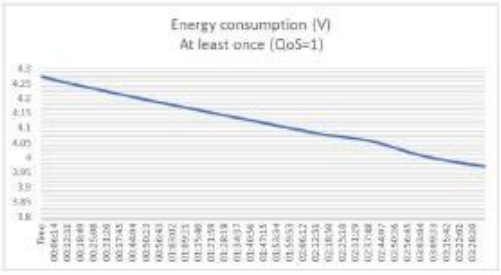


Fig. 8. Battery voltage level for MQTT "At least once" over TLS

Fig. 9 shows the battery voltage level for MQTT "Exactly once" over TLS.

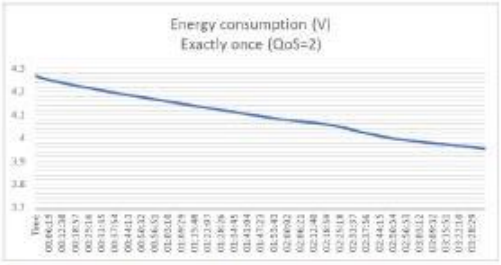


Fig. 9. Battery voltage level for MQTT "Exactly once" over TLS

The results of measurements are summarized in table II. Based on these results we can evaluate the difference in energy consumption of three MQTT protocol QoS levels over TLS. Less energy consumes "At least once (QoS1) over TLS – voltage drop 0.3026V. Most energy consumes "At most once (QoS0) over TLS – voltage drop 0.3228V and "Exactly once (QoS2)" over TLS consumes more energy than QoS1 and less than QoS0 - voltage drop 0.3176V.

TABLE II. Evaluation of the impact on energy consumption of MQTT protocol over TLS

MQTT QoS Level	Energy consumption	
	Voltage drop (V)	*Consumed time (hh:mm:ss)*
MQTT "At most once (QoS0)" over TLS	0.3228	03:34:45
MQTT "At least once (QoS1)" over TLS	0.3026	03:34:36
MQTT "Exactly once (QoS2)" over TLS	0.3176	03:34:45

VI. Conclusion

The energy consumption of MQTT protocol with various QoS over TLS levels is highly different. The main results of this paper are as follows:

1) The real time measured values for energy consumption securing MQTT over TLS are achieved with various QoS levels.

2) The results of energy consumption measurements when performing secure communication using MQTT protocol over TLS can be used to reliably predict energy consumption of three QoS levels:

- QoS "At least once (QoS1)" over TLS consumes less energy than the others two QoS levels (QoS0 over TLS and QoS2 over TLS),
- QoS "At most once (QoS0)" over TLS consumes more energy than the others two QoS levels (QoS1 over TLS and QoS2 over TLS),
- QoS "Exactly once (QoS2)" over TLS consumes 5 % more energy than QoS "At least once (QoS=1)" over TLS",
- QoS " At most once (QoS0)" over TLS consumes 6,7 % more energy than QoS "At least once (QoS1)" over TLS,
- QoS "Exactly once (QoS=2)" over TLS consumes 1,7 % less energy than QoS "At most once (QoS0) over TLS".

References

[1] Gartner Says Worldwide IoT Security Spending Will Reach $1.5 Billion in 2018. STAMFORD, Conn., March 21, 2018. Gartner, Inc. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2018-03-21-gartner-says-worldwide-iot-security-spending-will-reach-1-point-5-billion-in-2018 [Accessed February 07, 2019]

[2] R. Giambona, A. E.C. Redondi, M. Cesana, "Demonstrating MQTT+: An Advanced Broker for Data Filtering, Processing and Aggregation", In 21st ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM '18), October 28-November 2, 2018, Montreal, QC, Canada. ACM, New York, NY, USA. [Online]. Available: https://doi.org/10.1145/3242102.3243317

[3] S. B. Kenitar, S. Marouane, A. Mounir, "Evaluation of the MQTT Protocol Latency over Different Gateways", SCA2018, October 2018, Tetuan, Morocco. [Online]. Available: https://doi.org/10.1145/3286606.3286864

[4] M. Singh, R. MA, S. VL, and B. P, "Secure MQTT for Internet of Things (IoT)".2015 Fifth International Conference on Communication Systems and Network Technologies. IEEE Computer Society, 2015, pp. 746-751. DOI 10.1109/CSNT.2015.16

[5] MQTT Version 3.1.1 Plus Errata 01. OASIS Standard Incorporating Approved Errata 01. [Online]. Available: http://docs.oasis-

open.org/mqtt/mqtt/v3.1.1/errata01/os/mqtt-v3.1.1-errata01-os-complete.html [Accessed February 08, 2019]

[6] M. H. Amaran, M. S. Rohmad, L. H. Adnan, N. N. Mohamed, H. Hashim, "Lightweight Security for MQTT-SN". International Journal of Engineering & Technology, 7 (4.11) (2018) pp. 223-226

[7] M. Calabretta, R. Pecori, L. Veltri, "A Token-based Protocol for Securing MQTT Communications". 26th International Conference on Software, Telecommunications and Computer Networks (SoftCOM), 2018 . DOI:10.23919/SOFTCOM.2018.8555834

[8] A. L. Marra, F. Martinelli, P. Mori, A. Rizos, and A. Saracino, "Introducing Usage Control in MQTT". S. K. Katsikas et al. (Eds.): CyberICPS 2017/SECPRE 2017, LNCS 10683, Springer International Publishing AG 2018, pp. 35–43. https://doi.org/10.1007/978-3-319-72817-9_3

[9] G. Perrone, M. Vecchio, R. Pecori, and R. Giaffreda, "The Day After Mirai: A Survey on MQTT Security Solutions After the Largest Cyber-attack Carried Out through an Army of IoT Devices". In Proceedings of the 2nd International Conference on Internet of Things, Big Data and Security (IoTBDS 2017), p.p. 246-253. DOI: 10.5220/0006287302460253

[10] Eclipse Mosquitto. [Online]. Available: http://mosquitto.org/ [Accessed February 08, 2019]

[11] Erlang MQTT Client. [Online]. Available: https://github.com/emqtt/emqttc [Accessed February 08, 2019]

[12] Apollo. [Online]. Available: http://activemq.apache.org/apollo/ [Accessed February 08, 2019]

[13] Apache ActiveMQ Artemis. [Online]. Available: http://activemq.apache.org/artemis/ [Accessed February 08, 2019]

[14] J. Velez, R.Trafford, M. Pierce, B. Thomson, E. Jastrzebski, and B. Lau, "IEEE 1451-1-6: Providing Common Network Services over MQTT". IEEE Sensors Applications Symposium (SAS). 2018. DOI: 10.1109/SAS.2018.8336750

[15] S. H. Ramos , M. T. Villalba, and R. Lacuesta, "MQTT Security: A Novel Fuzzing Approach". Wireless Communications and Mobile Computing. Volume 2018, Hindawi, 11 pages. https://doi.org/10.1155/2018/8261746

[16] S. Zamfir, T. Balan, I. Iliescu, and F. Sandu, "A Security Analysis on Standard IoT Protocols". 2016 International Conference on Applied and Theoretical Electricity (ICATE). DOI: 10.1109/ICATE.2016.7754665

[17] M. Frustaci, P. Pace , G. Aloi, and G. Fortino, "Evaluating Critical Security Issues of the IoT World: Present and Future Challenges". IEEE Internet of things journal, vol. 5, No. 4, august 2018, pp. 2483-2495. DOI: 10.1109/JIOT.2017.2767291

[18] W. Ejaz, M. Naeem, A. Shahid, A. Anpalagan, and M. Jo, "Efficient Energy Management for the Internet of Things in Smart Cities". IEEE Communications Magazine, January 2017, pp. 84-91. DOI: 10.1109/MCOM.2017.1600218CM

[19] J. L. Espinosa-Aranda, N. Vallez, C. Sanchez-Bueno, D. Aguado-Araujo, G. Bueno, O. Deniz, "Pulga, a tiny open-source MQTT broker for flexible and secure IoT deployments". 1st IEEE Workshop on Security and Privacy in the Cloud, Florence (Italy), September 30, 2015

[20] IBM, "Qualities of service provided by an MQTT client". [Online]. Available: https://www.ibm.com/support/knowledgecenter/en/SSFKSJ_8.0.0/com.ibm.mq.dev.doc/q029090_.htm [Accessed February 8, 2019]

[21] 20-Nov-2018 OpenSSL 1.1.0j is now available, including bug and security fixes [Online]. Available: https://www.openssl.org/ [Accessed January 4, 2019]

7.2. Geriausio pranešimo apdovanojimas

IVUS 2019

BEST PAPER AWARD

In the 24th International Conference on Information Technology (IVUS 2019)

To

Edgaras Baranauskas

For the paper entitled

„Evaluation of the impact on energy consumption of MQTT protocol over TLS"

Dean of Vytautas Magnus University
Faculty of Informatics

Prof. Dr. Daiva Vitkutė-Adžgauskienė

VYTAUTAS MAGNUS UNIVERSITY

ktu kaunas university of technology

Kaunas Faculty

UNIVERSITÀ degli STUDI di CATANIA

Silesian University of Technology