Kauno technologijos universitetas

Informatikos fakultetas

Dviejų faktorių (2FA) skaitmeninio autentifikavimo metodas nuotolinei prieigai prie kritinės infrastruktūros sistemos

Baigiamasis magistro studijų projektas

Konstantinas Jurgilas

Projekto autorius

doc. Rasa Brūzgienė

Vadovė

Kaunas, 2021

Kauno technologijos universitetas

Informatikos fakultetas

Dviejų faktorių (2FA) skaitmeninio autentifikavimo metodas nuotolinei prieigai prie kritinės infrastruktūros sistemos

Baigiamasis magistro studijų projektas

6211BX008 Informacijos ir informacinių technologijų sauga

Konstantinas Jurgilas

Projekto autorius

doc. Rasa Brūzgienė

Vadovė

doc. Nerijus Morkevičius

Recenzentas

Kaunas, 2021

Kauno technologijos universitetas

Informatikos fakultetas

Konstantinas Jurgilas

Dviejų faktorių (2FA) skaitmeninio autentifikavimo metodas nuotolinei prieigai prie kritinės infrastruktūros sistemos

Akademinio sąžiningumo deklaracija

Patvirtinu, kad:

1. baigiamąjį projektą parengiau savarankiškai ir sąžiningai, nepažeisdama(s) kitų asmenų autoriaus ar kitų teisių, laikydamasi(s) Lietuvos Respublikos autorių teisių ir gretutinių teisių įstatymo nuostatų, Kauno technologijos universiteto (toliau – Universitetas) intelektinės nuosavybės valdymo ir perdavimo nuostatų bei Universiteto akademinės etikos kodekse nustatytų etikos reikalavimų;

2. baigiamajame projekte visi pateikti duomenys ir tyrimų rezultatai yra teisingi ir gauti teisėtai, nei viena šio projekto dalis nėra plagijuota nuo jokių spausdintinių ar elektroninių šaltinių, visos baigiamojo projekto tekste pateiktos citatos ir nuorodos yra nurodytos literatūros sąraše;

3. įstatymų nenumatytų piniginių sumų už baigiamąjį projektą ar jo dalis niekam nesu mokėjęs (-usi);

4. suprantu, kad išaiškėjus nesąžiningumo ar kitų asmenų teisių pažeidimo faktui, man bus taikomos akademinės nuobaudos pagal Universitete galiojančią tvarką ir būsiu pašalinta(s) iš Universiteto, o baigiamasis projektas gali būti pateiktas Akademinės etikos ir procedūrų kontrolieriaus tarnybai nagrinėjant galimą akademinės etikos pažeidimą.

Konstantinas Jurgilas

Patvirtinta elektroniniu būdu

Jurgilas, Konstantinas. Dviejų faktorių (2FA) skaitmeninio autentifikavimo metodas nuotolinei prieigai prie kritinės infrastruktūros sistemos. Magistro baigiamasis projektas / vadovė doc. Rasa Brūzgienė; Kauno technologijos universitetas, Informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informatikos inžinerija, Informacijos sistemos.

Reikšminiai žodžiai: kelių faktorių autentifikacija; kritinės infrastruktūros sistemos; *Kubernetes* valdymo skydelis.

Kaunas, 2021. 131 p.

Santrauka

Ypatingos svarbos (kitaip – kritinės) infrastruktūros sistemos apima svarbiausias visuomenės funkcionavimo sritis, reikalingas įvairių paslaugų teikimui. Šioms kritinėms sistemoms egzistuoja vis didesnė grėsmė tapti skaitmeninių nusikaltimų ir atakų taikiniu, kadangi jos yra kertinės tiek valstybės, tiek visuomenės atžvilgiu. Viena iš svarbiausių priežasčių, sukeliančių grėsmę šių sistemų saugumui, yra silpna nuotolinės prieigos kontrolė – sistemos naudotojo tapatybės valdymas.

Šiame darbe yra siūlomas ir tiriamas kelių faktorių autentifikacijos metodas, skirtas apsaugoti nuotolinę prieigą prie kritinės infrastruktūros sistemos – *Kubernetes* sistemos valdymo skydelio. Metodas yra pagrįstas *push notification* technologijos, skaitmeninių sertifikatų bei autentifikacijos užklausų autorizavimo sinergija, siekiant išpildyti autentifikacijos procesui keliamus saugos reikalavimus kritinėse infrastruktūros sistemose ir užtikrinti saugų naudotojo identiteto patvirtinimo procesą. Autentifikacijos metodas yra realizuojamas vieningo prisijungimo sistemos prototipe. Darbe yra tiriamos metodo kokybinės ir kiekybinės charakteristikos, o tyrimo rezultatai yra lyginami su kitų mokslininkų pristatomais metodais.

Metodo ir jį palaikančios sistemos eksperimentiniai tyrimai apėmė sistemos komponentų statinę kodo analizę; metodo kokybinių charakteristikų, kurios buvo suskirstytos į tris kategorijas - panaudojamumas (angl. *usability*), dislokavimas (angl. *deployment*) ir saugumas (angl. *security*) - tyrimą; *WEB* serverių konfigūracijos statinę ir dinaminę analizę; įsiskverbimo ir prieigos taškų greitaveikos testavimą; konfigūracijos ir autentifikavimo procesų atlikimo greičio tyrimą.

Darbo rezultatų aprobavimas:

1. Jurgilas, Konstantinas. Subjekto 2FA skaitmeninio autentifikavimo prie kritinės infrastruktūros informacinės sistemos struktūrizuotas vertinimas // Konferencijos "Lietuvos magistrantų informatikos ir IT tyrimai" darbai. Vilnius: Vilniaus universiteto leidykla. ISBN 978-609-07-0623-7. 2021, p. 23-33. DOI: https://doi.org/10.15388/LMITT.2021

Jurgilas, Konstantinas. Two-factor (2FA) digital authentication method for remote access to critical infrastructure system. Master's Final Degree Project / supervisor assoc. prof. Rasa Bruzgiene; Informatics Faculty, Kaunas University of Technology.

Study field and area (study field group): Informatics engineering, Information systems.

Keywords: multi-factor authentication; critical infrastructure systems; *Kubernetes* management dashboard.

Kaunas, 2021. 131 p.

Summary

The infrastructure systems of a high-importance (critical ones) encompass the most important areas and services that ensure the proper functions of a society. These critical systems are at risk of becoming targets of cyber-crimes and attacks because they are essential both at the level of state and society. One of the most prominent threats, posed to the security of these systems, is weak remote access control – the management of user identity.

This work proposes and analyses a multi-factor authentication method intended to protect remote access to critical infrastructure system – management dashboard of *Kubernetes* system. The method is based on the synergy of *push notification* technology, digital certificates, and authorization of authentication requests that aim to satisfy security requirements established for the process of authentication within critical infrastructure systems and to ensure secure verification of user identity. The authentication method is implemented in the prototype of a single-sign-on system. Research of qualitative and quantitative characteristics of the method as well comparative analysis of those results with methods, introduced in other papers and related with the process of subject's authorization, are presented in this work.

The experimental analysis of the method and its supporting single-sign-on system is based on the static analysis of system components; analysis of the method usability, deployment, and security qualitative characteristics; static and dynamic analysis of *WEB* servers' configuration; penetration and performance testing of the *API* endpoints; measurement of method configuration and authentication processes completion time.

Approbation of the work results:

1. Jurgilas, Konstantinas. Subjekto 2FA skaitmeninio autentifikavimo prie kritinės infrastruktūros informacinės sistemos struktūrizuotas vertinimas // Proceedings of the Conference "Lithuanian MSc Research in Informatics and ICT". Vilnius: Vilnius University Press. ISBN 978-609-07-0623-7. 2021, p. 23-33. DOI: https://doi.org/10.15388/LMITT.2021

Turinys

Lentelių sąrašas

Paveikslų sąrašas

Santrumpų ir terminų sąrašas

Santrumpos:

- *API* (angl. *Application Programming Interface*) – programinės įrangos sąsaja, teikianti tam tikrą funkcionalumą ar duomenų apsikeitimo kanalus, kuriais gali pasinaudoti programuotojai ar kitos sistemos;
- *AWS* (angl. *Amazon Web Services*) – vienas iš didžiausių debesijos paslaugų tiekėjų;
- *HTTP* (angl. *Hypertext Transfer Protocol*) – hipertekstų persiuntimo protokolas žiniatinklio duomenims persiųsti. Apibrėžia *HTTP* serverio ir kliento programos (dažniausiai naršyklės) sąveiką;
- *JSON* (angl. *JavaScript Object Notation*) – tekstinis ir lengvai suprantamas duomenų saugojimo ir perdavimo formatas, dažniausiai skirtas duomenų perdavimui tarp skirtingų paslaugų ar komponentų;
- *JWT* (angl. *JSON Web Token*) – atviras standartas, apibūdinantis kompaktišką būdą saugiai perduoti informaciją *JSON* formatu;
- *UML* (angl. *Unified Modeling Language*) – bendro naudojimo modeliavimo ir specifikacijų kalba, skirta programinės įrangos artefaktų kūrimui;
- *OWASP* (angl. *Open Web Application Security Project*) – tarptautinė ne pelno siekianti organizacija, dirbanti internetinių programų saugos srityje.

Terminai:

- Autentifikacija – subjekto tapatumo patikrinimo ir patvirtinimo procesas;
- Autorizacija – autentifikuotiems subjektams prieigos teisių prie resurso suteikimas pagal tam tikras nustatytas taisykles;
- *OAuth 2.0* – atviras standartas, skirtas naudotojų autorizacijos delegavimui;
- Konteineris (angl. *container*) – programinės įrangos, jos priklausomybių ir veikimo aplinkos vienetas, diegiamas konteinerių klasteryje;
- Atvaizdas (angl. *image*) – šablonas, kuris nurodo iš kokių sudedamųjų dalių ir kaip turi būti sukuriamas konteineris;
- Debesija (angl. *cloud*) - interneto paslaugų visuma, jungianti įvairiuose serveriuose esančius informacijos išteklius ir programinę įrangą, sudaranti sąlygas jais naudotis.

Įvadas

Ypatingos svarbos (kitaip – kritinės) infrastruktūros sistemos apima svarbiausias visuomenės funkcionavimo sritis, reikalingas įvairių paslaugų teikimui, įtraukiant transporto ir pašto, informacinių technologijų ir elektroninių ryšių, viešojo saugumo, valstybės valdymo, energetikos, užsienio reikalų, saugumo politikos sektorius.

Kritinėms sistemoms egzistuoja vis didesnė grėsmė tapti skaitmeninių nusikaltimų ir atakų taikiniu, kadangi jos yra kertinės tiek valstybės, tiek visuomenės atžvilgiu ir palaiko nenutrūkstamą šių sektorių teikiamų paslaugų vykdymą. Visuomenė yra priklausoma nuo šių paslaugų teikimo, tad bet koks jų sutrikdymas gali sukelti nepatogumų ar didelių neigiamų padarinių, pavyzdžiui 2015 metų gruodį įvykdytas kibernetinis išpuolis prieš Ukrainos elektros tiekimo tinklą, kuomet beveik ketvirtis milijono žmonių vidury žiemos liko be elektros energijos šešioms valandoms, arba 2010 metais Irane, Natanzo mieste esančios atominio kuro gamyklos užkrėtimas *Stuxnet* virusu.

Viena iš svarbiausių priežasčių, sukeliančių grėsmę šių sistemų saugumui, yra silpna nuotolinės prieigos kontrolė – sistemos naudotojo tapatybės valdymas. Siekiant sumažinti nesankcionuotos prieigos grėsmę, reikia tobulinti autentifikacijos procesą ir jame taikyti saugius identiteto patvirtinimo metodus.

Projekto naujumas ir aktualumas

Mokslinis darbo naujumas - pasiūlytas autentifikacijos metodas, pagrįstas *push notification* technologijos, skaitmeninių sertifikatų bei autentifikacijos užklausų autorizavimo sinergija, skirta valdyti nuotolinę prieigą prie kritinės infrastruktūros sistemos objekto.

Technologinis darbo naujumas - standartinio *Kubernetes* valdymo skydelio autentifikacijos mechanizmo pakeitimas nauju autentifikacijos sprendimu, kuris tenkintų kritinės infrastruktūros sistemoms keliamus saugos reikalavimus.

Tikslas ir uždaviniai

Šio darbo **mokslinė problema** yra autentifikavimo metodų, kurie tenkintų kritinės infrastruktūros sistemų keliamus saugos reikalavimus, trūkumas. **Darbo tikslas** - pasiūlyti dviejų faktorių skaitmeninio autentifikavimo ir autorizavimo metodą nuotolinei prieigai prie kritinės infrastruktūros objekto ir jį eksperimentiškai ištirti.

Darbo atlikimui yra keliami šie **uždaviniai**:

1. Išanalizuoti kritinės infrastruktūros sistemose taikomą autentifikacijos procesą bei jam taikomus saugos reikalavimus;
2. Išanalizuoti autentifikacijos metodus, taikomus *Kubernetes* konteinerių klasterio valdymo skydelyje;
3. Išanalizuoti egzistuojančius autentifikavimo metodus ir įvertinti jų tinkamumą apsaugant nuotolinę prieigą prie kritinės infrastruktūros sistemų objektų;
4. Suprojektuoti dviejų faktorių autentifikavimo ir autorizavimo metodą, kuris tenkintų kritinės infrastruktūros sistemoms keliamus saugos reikalavimus;
5. Suprojektuoti ir realizuoti vieningo prisijungimo sistemos prototipą, kuris taikytų siūlomą autentifikavimo metodą;

6. Suprojektuoti ir realizuoti tarpinio įgaliotojo serverio prototipą, kuris naudotų vieningo prisijungimo sistemos funkcionalumą ir apsaugotų nuotolinę prieigą prie *Kubernetes* valdymo skydelio;
7. Atlikti eksperimentinį pasiūlyto metodo veikimo tyrimą, įvertinant metodo realizacijos kokybines ir kiekybines charakteristikas.

Dokumento struktūra

Darbą sudaro keturi skyriai:

1. Kritinės infrastruktūros sistemų skaitmeninės autentifikacijos saugos analizė – Šiame skyriuje yra susipažįstama su kritinės infrastruktūros sistemomis, jų saugos problemomis bei taikomais skaitmeninio autentifikavimo procesais. Taip pat yra apžvelgiami egzistuojantys autentifikacijos metodai bei juos teikiantys produktai. Galiausiai, yra išanalizuojamas *Kubernetes* konteinerių klasterio valdymo skydelyje taikomas autentifikacijos metodas;
2. Kelių faktorių autentifikacijos metodo ir jį palaikančios sistemos projektas – Šiame skyriuje yra sudaroma problemos sprendimo aukšto lygio koncepcija, nustatomi sistemos realizacijai keliami funkciniai ir nefunkciniai reikalavimai bei sudaromas sistemos modelis;
3. Kelių faktorių autentifikacijos metodo ir jį palaikančios sistemos realizacija – Šiame skyriuje yra pateikiama informacija apie vieningos prisijungimo sistemos prototipą sudarančių komponentų realizacijos bei virtualios infrastruktūros detales;
4. Kelių faktorių autentifikacijos metodo ir jį palaikančios sistemos tyrimas – Šiame skyriuje yra ištiriamos ir įvertinamos siūlomo autentifikacijos metodo ir jį realizuojančios vieningo prisijungimo sistemos prototipo kokybinės ir kiekybinės charakteristikos.

Darbo rezultatų aprobavimas

1. Jurgilas, Konstantinas. Subjekto 2FA skaitmeninio autentifikavimo prie kritinės infrastruktūros informacinės sistemos struktūrizuotas vertinimas // Konferencijos "Lietuvos magistrantų informatikos ir IT tyrimai" darbai. Vilnius: Vilniaus universiteto leidykla. ISBN 978-609-07-0623-7. 2021, p. 23-33. DOI: https://doi.org/10.15388/LMITT.2021

1. Kritinės infrastruktūros sistemų skaitmeninės autentifikacijos saugos analizė

Prieš atliekant kritinės infrastuktūros sistemų skaitmeninės autentifikacijos saugos analizę, yra svarbu nustatyti šios analizės tikslus ir apibrėžti sąsajas su tolimesniais darbo etapais:

1. Susipažinti su kritinės infrastruktūros sistemomis ir jų saugos problemomis;
2. Išanalizuoti skaitmeninio autentifikavimo procesą bei jo taikymą valdant nuotolinę prieigą prie kritinės infrastruktūros sistemų;
3. Apžvelgti egzistuojančius autentifikacijos paslaugas teikiančius produktus;
4. Išanalizuoti *Kubernetes* konteinerių klasterio valdymo skydelyje taikomą autentifikacijos metodą.

1.1. Kritinių sistemų analizė

Informacinė infrastruktūra – tai įrangos, sistemų, taikomųjų programų bei palaikančių sistemų visuma, skirta sėkmingam informacinės visuomenės, kuri remiasi interneto teikiamomis paslaugomis, palaikymui [1]. Pagal šios infrastruktūros svarbą visuomenei bei jai keliamus techninius skaitmeninės saugos reikalavimus, ji yra skirstoma į kelias kategorijas [2]. Kiekviena žemesnio lygmens kategorija paveldi arba perrašo aukštesnio lygmens kategorijos saugos reikalavimus (t.y. antra kategorija paveldi trečios kategorijos reikalavimus):

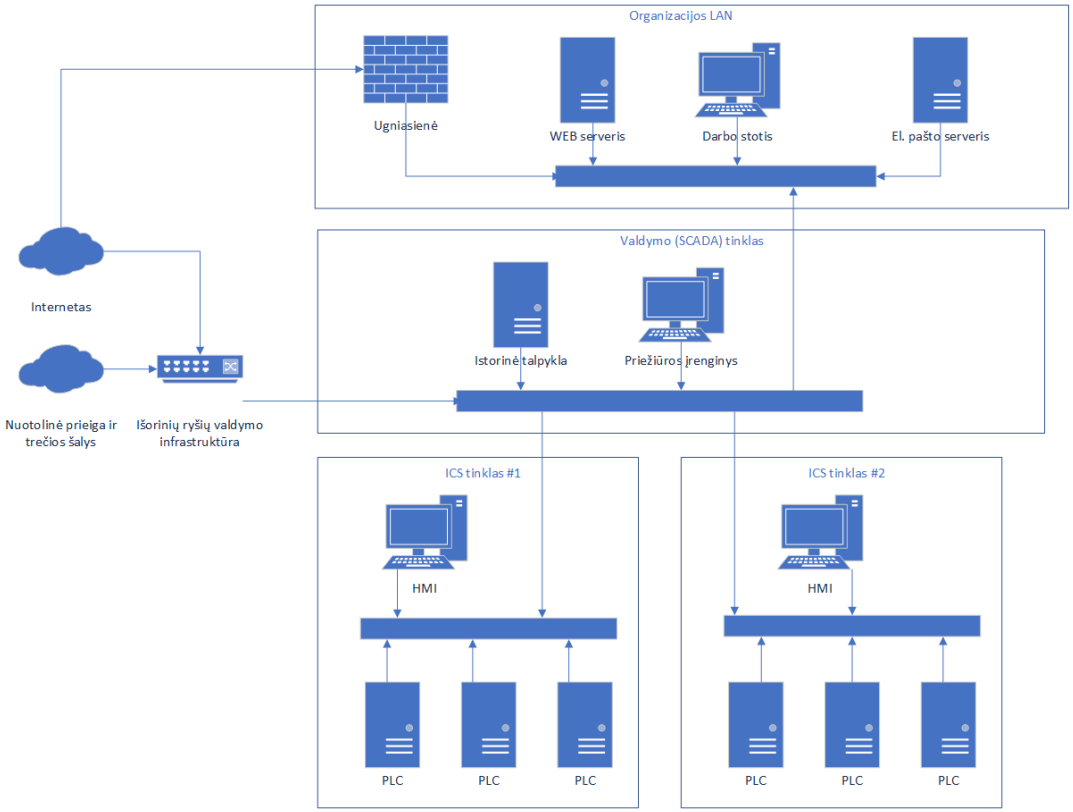
1. Ketvirtos kategorijos informacinė infrastruktūra – taikomos bazinės saugos priemonės: konfidencialumo užtikrinimas naudojant šifravimą, virtualųjį privatų tinklą, slaptažodis turi būti sudarytas iš raidžių, skaičių ir specialiųjų simbolių, slaptažodžiai negali būti saugomi ar perduodami atviru tekstu, neteisingai įvedus slaptažodį daugiau nei 5 kartus, paskyra yra užblokuojama. Pirmąkart jungiantis prie informacinės infrastruktūros, turi būti reikalaujama, kad naudotojas pakeistų numatytąjį slaptažodį;
2. Trečios kategorijos informacinė infrastruktūra – nuo šios kategorijos administratoriaus slaptažodis turi būti keičiamas ne rečiau kaip kas du mėnesius ir slaptažodį turi sudaryti ne mažiau kaip dvylika simbolių. Turi būti naudojamas *TLS* (angl. *Transport Layer Security*) standartas;
3. Antros kategorijos informacinė infrastruktūra – nuo šios kategorijos yra privaloma naudoti dviejų veiksnių tapatumo patvirtinimo priemones. Kiekviename audito duomenų įraše turi būti fiksuojama įvykio data/laikas, įvykio rūšis/pobūdis, duomenys ir rezultatai, susiję su įvykiu. Įvykus įtartinai veiklai, tai turi būti užfiksuojama audito įrašuose ir kuriamas pranešimas, kurį matytų administratorius;
4. Pirmos kategorijos informacinė infrastruktūra – nuo šios kategorijos neteisingai įvedus slaptažodį daugiau nei 3 kartus, paskyra yra užblokuojama. Mobilaus ryšio įrenginiuose privalo būti naudojamos centralizuotai valdomos ir atnaujinamos kenkimo programinės įrangos aptikimo, užkardymo ir stebėjimo priemonės;
5. Ypatingos svarbos informacinė infrastruktūra – remiantis šiuo metu (2021-03-19) galiojančiu įstatymu, pagal keliamus reikalavimus ši kategorija sutampa su pirmos kategorijos informacine infrastruktūra.

Ypatingos svarbos arba kritinės infrastruktūros sistemos - tai sistemos, sudarytos iš techninės ir programinės įrangos, kurių visuma teikia tam tikras globalias funkcijas ir paslaugas, kurių sutrikimai gali sukelti didelę žalą nacionaliniam saugumui, ekonomikos stabilumui ar žmonių sveikatai ir gerbūviui [3]. Kritinės infrastruktūros sistemos yra naudojamos energetikos, informacijos ir

informacinių technologijų, vandens tiekimo, sveikatos apsaugos, finansų, transporto, chemijos, kosmoso tyrimų bei nacionalinio saugumo sektoriuose.

Šios sistemos yra labai svarbios, kadangi jos palaiko nenutrūkstamą šių sektorių teikiamų paslaugų vykdymą. Visuomenė yra priklausoma nuo šių paslaugų teikimo, tad bet koks jų sutrikdymas gali sukelti nepatogumų, pavyzdžiui elektros energijos ar vandens tiekimo sustabdymas. Visgi sutrikus šioms sistemoms, padariniai gali būti dar didesni. Pavyzdžiui, sutrikus atominės elektrinės veikimui gali įvykti sprogimas, kuris lemtų radioaktyvių dalelių išmetimą į atmosferą. Atsižvelgus į sutrikimų keliamas grėsmes, kritinės infrastruktūros sistemoms yra keliami labai aukšti veikimo patikimumo standartai.

Kritinės infrastruktūros sistemos dažnai yra realizuojamos naudojant *SCADA* (angl. *Supervisory Control and Data Acquisition*) valdymo sistemos architektūrą [4]. *SCADA* yra paskirstytos architektūros sistema, kurią sudaro įvairūs komponentai – priežiūros kompiuteriai, programuojami loginiai valdikliai, nutolę terminaliniai įrenginiai ir įvedimo/išvedimo įrenginiai. 1 pav. yra pateikta standartinė *SCADA* sistemos architektūra.



1 pav. Standartinė *SCADA* sistemos architektūra [5]

Šie komponentai tarpusavyje komunikuoja naudodami bendrą tinklą. Taip pat ir pačios *SCADA* sistemos vis dažniau yra prijungiamos į bendrą interneto tinklą, siekiant praplėsti jų veikimo funkcijas bei pritaikymą.

Kritinės infrastruktūros sistemos palaipsniui pereina į debesiją. Debesijos teikiami privalumai yra labai svarbūs kritinėms sistemoms. Pertekliškumas, patikimumas, prieinamumas bei plėtimas yra labai svarbios savybės, kurias teikia debesijos ištekliai. Yra pastebima tendencija [6], jog vis didesnė dalis kritinės infrastruktūros sistemų yra perkeliamos iš vidinių patalpų (angl. *on-premise*) į debesiją.

Visgi perkeliant kritinės infrastruktūros sistemas į debesiją atsiranda tam tikrų iššūkių. Senas sistemas, turinčias monolitinę architektūrą, yra sunkiau perkelti į debesiją, kadangi jų architektūra nėra pritaikyta išnaudoti debesijos teikiamus privalumus. Modernios, debesijos ištekliams pritaikytos sistemos dažniausiai vadovaujasi dvylikos faktorių metodologija [7]. Debesijai pritaikytos programos yra konteinerizuojamos naudojant tokius įrankius kaip *Docker*, kurie leidžia virtualizuoti sistemų veikimo aplinką. Konteinerizuotos programos tuomet yra valdomos konteinerių orkestravimo sistemų, pvz. *Kubernetes* ar *OpenShift*.

Perėjimas į debesiją teikia daugelį privalumų, tačiau taip pat kelia ir naujus iššūkius: tarp jų - saugos grėsmes, kurios kyla kritinių infrastruktūrų sistemoms.

1.2. Kritinių sistemų saugos analizė

Palaipsninis kritinių infrastruktūrų sistemų įjungimas į bendrą interneto tinklą lemia, jog kritinės infrastruktūros sistemos vis dažniau tampa kibernetinių nusikaltėliu taikiniu ir tai kelia dideles saugos grėsmes. 2019 metų Jungtinių Amerikos Valstijų Nacionalinės Žvalgybos strategijos ataskaitoje [8] perspėjama, kad kibernetinės grėsmės kelia riziką viešajai gerovei, saugumui bei klestėjimui, kadangi informacinės technologijos, plačiai naudojamos visuomenės, yra integruotos į kritinę infrastruktūrą [9]. Ataskaitoje taip pat yra iškeliami nacionaliniai saugumo tikslai ir vienas iš jų - apsaugoti kritinę infrastruktūrą.

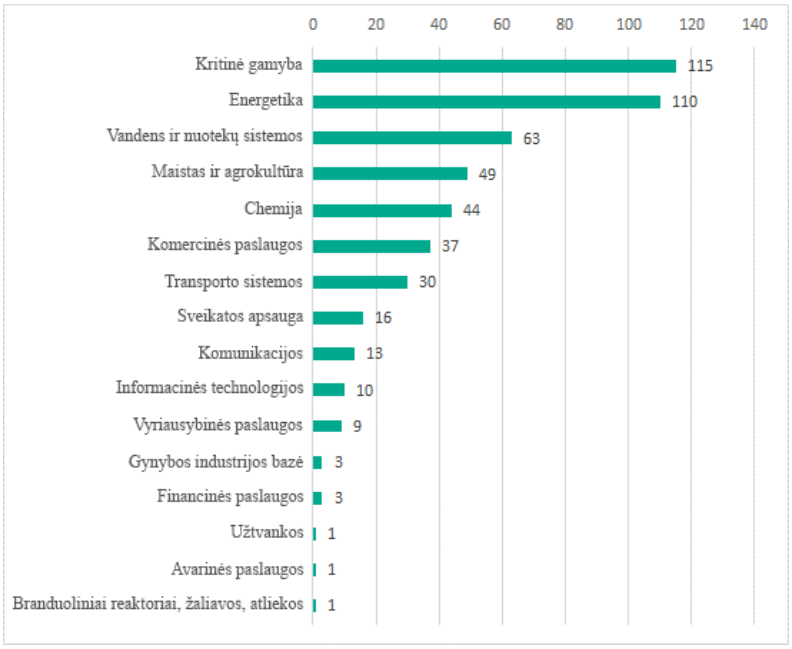
Kibernetinės atakos, nukreiptos į kritinę infrastruktūrą, jau keletą kartų privertė sunerimti visą pasaulį. 2010 metais Irane, Natanzo mieste esanti atominio kuro gamykla buvo užkrėsta *Stuxnet* virusu. Šis kirminas buvo sukurtas siekiant pažeisti valdymo motorus, dažniausiai naudojamus urano sodrinimo centrifugose. Virusui pavyko laikinai sustabdyti 1000 centrifugų darbą ir taip priversti gamyklą sustabdyti darbą [9].

2015 metų gruodį Ukraina patyrė kibernetinį išpuolį prieš elektros tiekimo tinklą. Nusikaltėliai įsibrovė į trejas energetikos kompanijas ir išjungė šių kompanijų energijos generatorius trijuose Ukrainos regionuose. Atakos metu beveik ketvirtis milijono žmonių vidury žiemos liko be elektros energijos šešioms valandoms. Atlikus tyrimą, buvo nustatyta, jog nusikaltėliai panaudojo *BlackEnergy3* virusą, kuris manomai buvo platinamas naudojant elektroninių pranešimų klastotes (angl. *phishing*).

2017 metais kibernetiniai teroristai perėmė nuotolinę prieigą prie darbo stoties, kuri manomai buvo Saudo Arabijoje [9]. Teroristai panaudojo naują virusą, kuriam buvo duota pravardė *Triton*, kuris suteikė galimybę perimti jėgainės saugos prietaisų sistemos valdymą. Tyrėjai teigia, jog tai buvo sabotažas, kurio metu buvo siekiama sukelti sprogimą, išjungiant saugumo sistemas, kurios buvo skirtos išvengti katastrofinių incidentų. Ataskaitų duomenimis, tik programavimo klaida lėmė, jog buvo išvengta katastrofos. Įrodymai rodo, jog virusas taip pat buvo platinamas naudojant apgavikiškus laiškus.

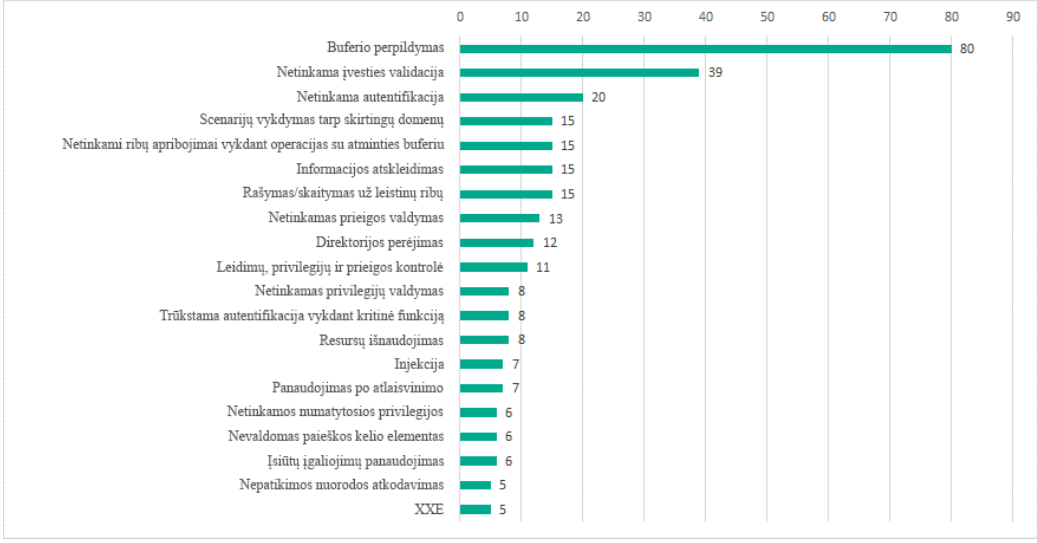
Kibernetinio saugumo statistikoje yra pastebima tendencija, jog kibernetinių atakų, nukreiptų prieš kritinės infrastruktūros sistemas, ir aptiktų pažeidžiamų sistemų komponentų skaičiai auga. 2018 metų *Kaspersky* kompanijos sudarytoje saugumo rizikų, skirtų industrinėms automatizavimo sistemoms, ataskaitoje [10], kuri apibendrina ICS-CERT internetinėje svetainėje [11] pateikiamus duomenis, nurodoma, jog dažniausiai atakuojamos yra energetikos, vandens tiekimo bei kritinės

gamybos sistemos. 2 pav. yra pateikiama statistika apie 2018 metais aptiktus pažeidžiamus produktus skirtinguose sektoriuose.



2 pav. 2018 metais aptiktų pažeidžiamų produktų skaičiai skirtingose sektoriuose [10]

Ataskaitoje taip pat yra nurodomas aptiktų pažeidžiamumų tipų pasiskirstymas. Ši statistika yra pateikiama 3 pav.

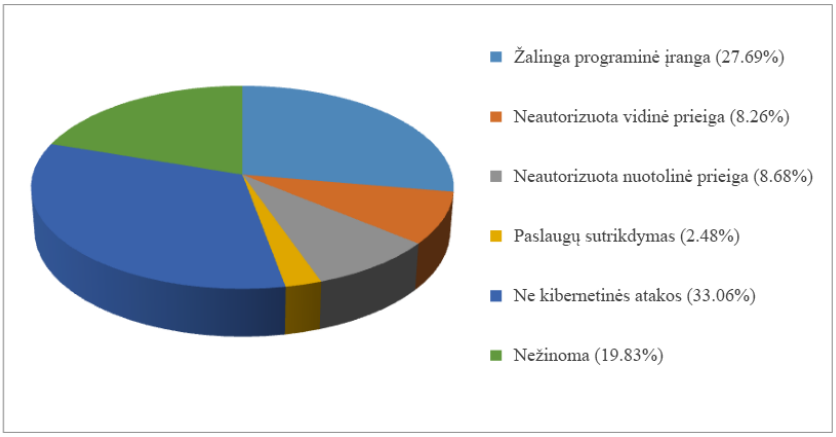


3 pav. 2018 metais aptiktų pažeidžiamumų tipų pasiskirstymas [10]

Ataskaitoje nurodoma, jog dažniausiai aptinkami pažeidžiamumai yra buferio perpildymas ir netinkama įvesčių validacija. Šioje statistikoje taip pat nurodoma, jog 16% aptiktų pažeidžiamumų yra susiję su autentifikacijos problemomis (netinkama autentifikacija, autentifikacijos apėjimu, trūkstama autentifikacija vykdant kritines funkcijas) ir prieigos kontrolės problemomis (neteisingi numatytieji leidimai, neteisingas privilegijų valdymas, įgaliojimų valdymas).

Kitame 2017 metais vykdytame tyrime [12] buvo analizuojamos 242 kibernetinės atakos prieš kritinę infrastruktūrą. Šio tyrimo metu buvo renkama statistika apie atakų pasiskirstymą skirtinguose

sektoriuose, incidentų dažnį, atakų tikslą bei naudotus metodus. 4 pav. pateikiama statistika apie kibernetinių atakų vykdymo metodus.



4 pav. Kibernetinių atakų vykdymo metodai [12]

Statistikoje nurodoma, jog iš 242 tirtų kibernetinių atakų prieš kritinę infrastruktūrą, 27.69% atakų buvo įvykdomos dėl virusų, 8.26% dėl neautorizuotos vidinių darbuotojų prieigos, o 8.68% - dėl neautorizuotos nuotolinės prieigos.

Istoriškai kritinės infrastruktūros sistemos dažnai buvo projektuotos dirbti tik vidiniuose tinkluose ir būti izoliuotos nuo viešųjų tinklų. Visgi perėjimas į debesiją, poreikis dalintis informacija su skirtingais verslo padaliniais bei turėti nuotolinę prieigą prie šių sistemų lėmė kritinių sistemų prijungimą į bendrą interneto tinklą. Šis prijungimas suteikė ne tik privalumų, bet ir didelių trūkumų, kadangi sistemoms, prijungtoms prie interneto, yra didesnė rizika tapti pažeidžiamoms ir tapti kibernetinių nusikaltėlių taikiniu.

Nusikaltėliams gavus prieigą prie kritinės infrastruktūros sistemų, atakų padariniai gali būti labai nuostolingi ir sukelti daug žalos. Atsižvelgus į statistikos duomenis yra matoma, jog nuotolinė prieiga yra dažnas nusikaltėlių taikinys. Prastai realizuoti nuotolinės prieigos mechanizmai prie kritinių sistemų gali lemti, jog nusikaltėliai gali gauti nesankcionuotą prieigą ir atlikti žalingus veiksmus. Tad vienas iš svarbiausių saugumo tikslų yra užtikrinti, jog tik sankcionuoti asmenys nuotoliniu būdu galėtų valdyti sistemų veiklą.

1.3. Autentifikacijos proceso analizė

Autentifikacija – tai subjekto tapatumo patikrinimo ir patvirtinimo procesas. Prieš naudojantis sistema, naudotojas turi pateikti bei įrodyti savo tapatybę. Autentifikacijos proceso metu yra nurodomas tam tikras naudotojo identifikatorius, pvz. naudotojo vardas, darbuotojo kodas ar el. pašto adresas. Tuomet naudotojas pateikia privačius identitetą patvirtinančius duomenis, pvz. slaptažodį. Jei pateikti privatūs duomenys yra teisingi, naudotojo identitetas yra patvirtinimas ir naudotojas gali sėkmingai naudotis sistemos teikiamomis funkcijomis.

Autentifikacijos procesas yra labai svarbus kompiuterinėse sistemose, kadangi šio proceso metu yra vienareikšmiškai nustatomas sistemos naudotojo identitetas. Jei autentifikacijos procesas yra įvykdomas sėkmingai, sistema pasitiki proceso metu gautu rezultatu ir identifikuoja sistemos naudotoją tik pagal proceso metu gautą informaciją - autentifikacijos kontekstą. Jei autentifikacijos

procesas bus pažeistas, egzistuoja rizika, jog sistema neteisingai identifikuos bei autentifikuos jos naudotojus.

Projektuojant ir realizuojant autentifikavimo sistemas yra labai svarbu užtikrinti jų saugą. Įvairūs pažeidžiamumai, esantys autentifikacijos procese, gali lemti, jog piktavaliai asmenys gali apeiti saugos mechanizmus ir gauti neteisėtą prieigą prie sistemos arba gauti neteisėtą prieigą prie kitų sistemos naudotojų paskyrų. Išnaudojus autentifikacijos proceso pažeidžiamumus, galimos įvairios pasekmės – neteisėta prieiga prie privačios informacijos, neteisėtų veiksmų vykdymas ar sistemos veiklos perėmimas.

Privatūs autentifikavimo duomenys, kuriuos naudotojas pateikia sistemai, yra vadinami faktoriais. Faktoriai yra skirstomi į tris grupes:

- Žinių faktorius – tai, ką naudotojas žino – pavyzdžiui, slaptažodis, PIN kodas;
- Nuosavybės faktorius – tai, ką naudotojas turi – pavyzdžiui, kortelė, išmanusis telefonas ar žetonų generatorius;
- Biometrinis faktorius – tai, kas naudotojas yra – pavyzdžiui, biometrinė informacija ar elgsenos šablonai.

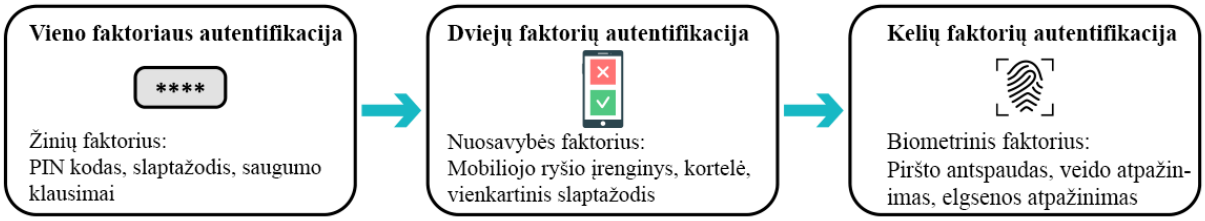
Tradiciškai sistemose yra taikomas vienintelis žinių faktorius - slaptažodis. Taikant tokį autentifikavimo mechanizmą naudotojas, norėdamas prisijungti prie sistemos, turi pateikti tik naudotojo identifikatorių ir su juo susijusį slaptažodį. Vis dėlto naudotojai yra linkę naudoti lengvai atspėjamus, paprastos struktūros slaptažodžius bei tuos pačius slaptažodžius naudoti keliose sistemose. Slaptažodžiai taip pat dažnai yra užrašomi ant lapukų ir priklijuojami lengvai prieinamose vietose, pvz. ant monitoriaus. [13] šaltinyje nurodoma, jog internetinėse sistemose saugomi slaptažodžių duomenys yra dažnai pavogiami. Šaltinyje taip pat pateikiama statistika bei faktai, jog 50% elektroninių pranešimų klastočių (angl. s*pear phishing*) [14] atakų yra sėkmingos, 75% žmonių naudoja tuos pačius slaptažodžius skirtingose sistemose bei, kad slaptažodžių sudėtingumo taisyklės nėra labai veiksmingos. Kuomet naudotojai naudoja tuos pačius slaptažodžius keliose skirtingose sistemose, atsiranda rizika, jog atskleidus slaptažodį vienoje sistemoje, bus gauta prieiga ir prie kitų sistemų paskyrų.

Jungtinių Valstijų Nacionalinis Standartų ir Technologijų institutas (angl. *NIST*) yra išleidęs publikaciją [15], kurioje yra pateikiama skaitmeninių identifikavimo paslaugų realizavimo politika ir gairės. Nors šis dokumentas yra skirtas federalinių agentūrų sistemoms, tačiau jame esančios rekomendacijos gali būti pritaikomos daugelyje sistemų. Publikacijoje yra pateikiami saugos reikalavimai, skirti identifikavimo proceso saugos užtikrinimui. Didelis dėmesys yra skiriamas slaptažodžių reikalavimams. Paskyrų slaptažodžių ilgis turi būti ne mažesnis nei 8 simboliai. Naudotojams taip pat turėtų būti draudžiama naudoti lengvai atspėjamus ir dažnai pasitaikančius slaptažodžius (pvz. „pass123“ ar sistemos pavadinimas) [16]. Gairėse taip pat didelis dėmesys yra skiriamas slaptažodžių saugojimui. Slaptažodžiai negali būti saugomi atviru tekstu – turi būti saugoma ne trumpesnė nei 32 bitų slaptažodžio santrauka panaudojant atsitiktinį „sūdymą“ (angl. *salt*). Turi būti naudojama *PBKDF2* raktų išvedimo funkcija, naudojant *SHA-2* ar *SHA-3* algoritmus. Taip pat slaptažodžiai turėtų būti saugomi užšifruotoje *HMAC* santraukoje.

Tačiau ne vien neatsargūs naudotojai yra kalti dėl slaptažodžių saugos problemų. Slaptažodžiais grįsta autentifikacija yra jautri brutalios jėgos atakoms, žiūrėjimo per petį atakoms, žodyno atakoms

ir socialinės inžinerijos atakoms. Šios slaptažodžių saugumo rizikos priverčia ieškoti naujų priemonių ir metodų, kurie leistų užtikrinti saugesnį autentifikavimo procesą.

Autentifikacijos procesas buvo patobulintas sukuriant dviejų faktorių autentifikacijos procesą. Šio proceso metu naudotojas, norėdamas patvirtinti savo tapatybę, turi pateikti du skirtingus faktorius - pavyzdžiui, pateikti statinį slaptažodį ir žetoną, kuris buvo sugeneruotas naudojant slaptažodžių generatorių. Vėliau šis metodas buvo apibendrintas ir pavadintas kelių faktorių autentifikacija, kuomet naudotojas turi pateikti du ar daugiau faktorių, norėdamas patvirtinti savo tapatybę. 5 paveikslėlyje yra pateikiama kelių faktorių autentifikacijos proceso evoliucija.



5 pav. Autentifikacijos proceso evoliucija [17]

Kelių faktorių autentifikacijos proceso tikslas – sukurti sluoksniuotą apsaugos mechanizmą, kuris leistų užtikrinti, jog neautorizuoti asmenys negalėtų gauti prieigos prie sistemų ir neteisėtai naudotis jų funkcijomis. Net jei vienas faktorius yra sukompromituojamas, pavyzdžiui, yra perimamas naudotojo slaptažodis, piktavalis negalės gauti prieigos prie sistemos nepateikęs ir kitų reikiamų faktorių, pavyzdžiui, autentifikacijos pavirtinimo naudojant susietą mobilųjį įrenginį. Kelių faktorių panaudojimas apsunkina sistemų naudotojų paskyrų perėmimą bei neteisėtos prieigos prie sistemų gavimą naudojant pavogtas paskyras.

1.4. Autentifikacijos veiksnių analizė

Šiame poskyryje bus apžvelgiami faktoriai, taikomi autentifikacijos proceso metu.

1.4.1. *SMS* žinutės

SMS žinutės yra vienas iš labiausiai paplitusių dviejų faktorių autentifikacijos metodų. Naudotojas, susiejęs savo paskyrą su aktyviu mobilaus ryšio vartotojo numeriu, autentifikacijos proceso metu gauna *SMS* žinutę, kurioje yra nurodomas 5 - 10 skaitmenų kodas, kurį reikia įvesti į prisijungimo formą, norint sėkmingai užbaigti autentifikacijos procesą. 6 pav. yra pateikta pavyzdinė *SMS* žinutė su prisijungimo kodu.

6 pav. *SMS* žinutės su prisijungimo kodu pavyzdys [18]

Sistemos, kuri autentifikacijos proceso metu naudoja *SMS* žinutes, naudojimas dažniausiai susideda iš dviejų procesų: mobilaus ryšio vartotojo numerio registracijos ir prisijungimo.

Mobiliojo telefono numerio registracijos procesas susideda iš šių žingsnių:

- Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
- Naudotojas paskyros nustatymuose nurodo galiojantį mobiliojo telefono numerį;
- Unikalus vienkartinis kodas yra sugeneruojamas ir nusiunčiamas į naudotojo telefoną;
- Naudotojas įveda gautą kodą į sistemą, taip patvirtindamas dviejų faktorių autentifikacijos aktyvavimą.

Prisijungimo procesas susideda iš šių žingsnių:

- Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
- Unikalus vienkartinis kodas yra sugeneruojamas ir nusiunčiamas į naudotojo telefoną;
- Naudotojas prisijungimo lange įveda gautąjį vienkartinį kodą;
- Jei kodas yra teisingas, naudotojas yra sėkmingai autentifikuojamas ir prijungiamas prie sistemos.

SMS žinutėmis paremto autentifikavimo metodo privalumai:

- Naudotojams šis metodas yra patogus, kadangi dauguma naudoja mobiliuosius įrenginius;
- Metodą realizuoti yra santykinai pigu.

SMS žinutėmis paremto autentifikavimo metodo trūkumai [19]:

- Metodas priklauso nuo mobiliojo ryšio būsenos, t.y. neturint telefono ryšio ar keliaujant užsienyje *SMS* žinutė su kodu yra potencialiai neatsiunčiama;
- Pametus telefoną kartu su *SIM* kortele, autentifikacija yra negalima;
- *GSM* ryšio terpė yra nepatikima ir galimas kodų perėmimas;
- Kodas gali būti perimamas telefone veikiančios žalingos *trojan* programinės įrangos;
- Prisijungimo kodas gali būti matomas net ir užrakinus telefoną, jei yra įjungta užrakto lango pranešimų funkcija;

– Metodas yra jautrus socialinės inžinerijos atakoms, kuomet nusikaltėliai gali gauti naujas *SIM* korteles su susietu aukos mobiliuoju numeriu. *SMS* žinutės tuomet bus siunčiamos į nusikaltėlio *SIM* kortelę, o aukos mobilusis įrenginys bus atjungtas nuo mobiliojo ryšio.

NIST organizacija 2016 metų liepos gale išleido publikaciją [20], kurioje nurodoma, jog dviejų faktorių autentifikavimo metodai, paremti *SMS* žinučių siuntimo principu, yra skelbiami nebenaudojamais (angl. *deprecated*).

1.4.2. El. paštas

El. paštas taip pat gali būti naudojamas dviejų faktorių autentifikacijos proceso metu. Taip kaip ir *SMS* žinutės metodo atveju, vienkartinis 5 - 10 simbolių kodas yra nusiunčiamas į naudojo el. pašto dėžutę. Naudotojui kodas gali būti pateikiamas keliais būdais:

– Kaip tekstas – naudotojas turi nukopijuoti kodą į prisijungimo formą;
– Kaip nuoroda – naudotojas turi paspausti nuorodą, norėdamas užbaigti prisijungimo procesą.

7 pav. yra pateiktas el. laiško pavyzdys, kuomet kodas yra paslepiamas nuorodoje, kurią naudotojas turi paspausti.

7 pav. El. laiško pavyzdys [35]

Prisijungimo procesas susideda iš šių žingsnių:

– Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
– Unikalus vienkartinis kodas yra sugeneruojamas ir nusiunčiamas į naudotojo el. pašto dėžutę;
– Naudotojas prisijungimo lange įveda gautąjį vienkartinį kodą arba el. laiške paspaudžia prisijungimo nuorodą;
– Jei kodas yra teisingas, naudotojas yra sėkmingai autentifikuojamas ir prijungiamas prie sistemos.

El. paštu paremto autentifikavimo metodo privalumai:

– Naudotojai gali gauti el. laiškus keliuose įrenginiuose (kompiuteriuose, mobiliuosiuose įrenginiuose);
– Metodą realizuoti yra santykinai pigu ir paprasta.

El. paštu paremto autentifikavimo metodo trūkumai:

– El. laiškai kartais nėra nusiunčiami sėkmingai;
– Nusikaltėliai gali perimti aukos el. pašto dėžutės paskyrą.

1.4.3. Vienkartiniai slaptažodžiai

Vienkartiniai slaptažodžiai (angl. *one-time passwords*) yra slaptažodžiai, naudojami autentifikacijos proceso metu, kurie gali būti panaudoti tik vieną kartą arba kurie galioja tam tikrą fiksuotą laiko tarpą. Vienkartinių slaptažodžių generavimui yra naudojami keli skirtingi algoritmai – *HMAC* [21] ar *TOTP* [22].

TOTP (angl. *Time-based One-Time Password*) algoritmas vienkartinį slaptažodį generuoja pasitelkdamas specifinę kriptografinę funkciją, kuri kaip įvestis priima bendrą paslaptį (angl. *secret*) ir dabartinio laiko žymą. Viena iš naudojamų kriptografinių funkcijų – *SHA-256*.

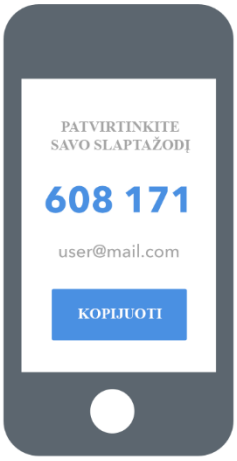
Sistemos, kuri autentifikacijos proceso metu naudoja vienkartinius slaptažodžius, naudojimas dažniausiai susideda iš dviejų procesų: slaptažodžių generatoriaus registracijos ir prisijungimo.

Slaptažodžių generatoriaus registracijos procesas susideda iš šių žingsnių:

– Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
– Dviejų faktorių autentifikacijos aktyvavimo lange yra pateikiama bendra paslaptis, kuri į mobilųjį įrenginį yra įvedama rankiniu būdu arba naudojant *QR* kodų skaitytuvą;
– Bendra paslaptis – raktas – yra išsaugoma į *TOTP* algoritmą realizuojančią mobiliąją programą;
– Naudotojui yra aktyvuojama dviejų faktorių autentifikacija.

Prisijungimo procesas susideda iš šių žingsnių:

– Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
– Jei prisijungimo duomenys teisingi, naudotojas yra nukreipiamas į kitą formą, kurioje yra prašoma įvesti vienkartinį slaptažodį, kurį sugeneruoja mobilusis įrenginys. 8 pav. yra pateikta vienkartinio slaptažodžio iliustracija;
– Sistema patikrina kodą, naudodama bendrą paslaptį, ir jei kodas yra teisingas, naudotojas yra sėkmingai autentifikuojamas bei prijungiamas prie sistemos.



8 pav. Vienkartinio slaptažodžio pavyzdys [35]

TOTP paremto autentifikavimo metodo privalumai:

- Kodai yra generuojami naudojant bendrą paslaptį ir dabartinio laiko žymę, tad naudotojas gali gauti teisingus kodus neturėdamas mobiliojo ryšio;
- Paslaptis yra saugoma mobiliajame įrenginyje ir nėra siunčiama tinklu, tad ji negali būti perimama.

TOTP paremto autentifikavimo metodo trūkumai:

- Naudotojas negali autentifikuotis, jei įrenginys yra išjungtas, pvz. pasibaigia baterija;
- Reikalinga užtikrinti mobiliojo įrenginio ir sistemos serverio laiko serviso sinchronizaciją;
- Perėmus bendrą paslaptį, piktavaliai gali patys generuoti kodus;
- Galima brutalios jėgos ataka, kuomet nėra tikrinamas prisijungimo kartų skaičius.

Egzistuoja keletas skirtingų vienkartinių slaptažodžių realizacijų. Viena iš jų – *RSA* raktai, kurių vienkartiniai slaptažodžiai dar vadinami žetonais. *RSA* autentifikacija yra grįsta dviem faktoriais – slaptažodžiu/*PIN* kodu ir autentifikatoriumi. Autentifikatorius gali būti fizinis įrenginys ar programinė įranga. Tiek fizinis įrenginys, tiek programinė įranga yra susiejama su konkrečia sistemos paskyra. Prisijungimo metu, įvedus slaptažodį/*PIN* kodą, naudotojas taip pat turi įvesti vienkartinį žetoną, kuris yra generuojamas tam tikru laiko intervalu (apie 60 sekundžių) naudojant vidinį laikrodį ir gamintojo sugeneruotą vidinį raktą. Kiekvienas žetonas turi skirtingą vidinį raktą, kuris yra įkeliamas į įrenginį susiejimo metu.

Verslo įmonėse dažnai yra naudojami fiziniai įrenginiai, kurie generuoja žetonus. 9 pav. yra pavaizduotas fizinis žetonų generatorius.

9 pav. *RSA* žetonų generatorius [23]

Fizinių žetonų privalumai:

- Tai yra visiškai nepriklausomas įrenginys, kuriam nereikia mobiliojo ar *Wi-Fi* ryšio.

Fizinių žetonų trūkumai:

- Įrenginiai yra pakankamai brangūs;
- Egzistuoja rizika, jog įrenginiai gali būti pavogti arba pamesti.

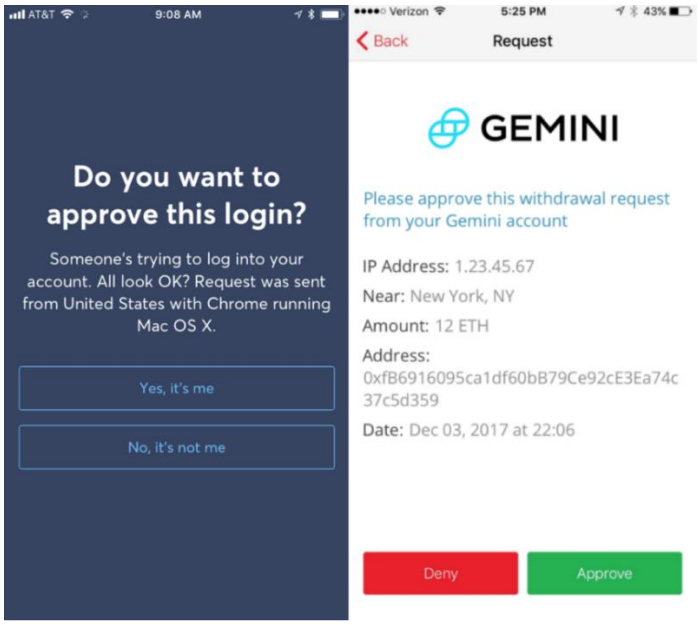
Vietoje fizinių žetonų gali būti naudojami programiniai (angl. *soft*) žetonai. Naudojant šį metodą, į kompiuterį ar mobilųjį įrenginį yra diegiama programinė įranga, kuri generuoja žetonus. 10 pav. yra pateiktas *RSA* programinis žetonų generatorius.

10 pav. *RSA* programinis žetonų generatorius [24]

Programinių žetonų generatoriaus privalumai ir trūkumai atitinka *TOTP* metodo privalumus ir trūkumus.

1.4.4. Mobiliojo ryšio įrenginys

Dviejų faktorių autentifikacijos proceso metu mobiliojo ryšio įrenginiai gali būti panaudoti ne tik kaip vienkartinių slaptažodžių ar žetonų generatoriai, bet ir realizuojant *push authentication* autentifikacijos metodą. Šio metodo metu naudotojai į susietą mobiliojo ryšio įrenginį gauna pranešimą/užklausą, kurioje gali patvirtinti ar uždrausti prisijungimą prie paskyros tam tikroje sistemoje. 11 pav. yra pateikiamas *push authentication* užklausos pavyzdys.



11 pav. *Push authentication* užklausos pavyzdys [25]

Sistemoje, taikančioje *push authentication* metodą, prisijungimo procesas yra sudarytas iš šių žingsnių:

- Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
- Į susietą naudotojo mobiliojo ryšio įrenginį yra nusiunčiamas *push* pranešimas, kuris yra prisijungimo užklausa. *Push* žinučių siuntimo metodologija yra apibūdinama [26] ir [27] šaltiniuose;
- Užklausoje yra nurodomas sistemos vardas, operacinė sistema ir naršyklė, kuri atlieka autentifikaciją, užklausos prašytojo lokacija ir užklausos data bei laikas;

– Naudotojas, atsižvelgdamas į pranešime pateiktą informaciją, patvirtina prisijungimo užklausą ir yra automatiškai prijungiamas prie sistemos.

Push authentication paremto autentifikavimo metodo privalumai:

– Naudotojams šis metodas yra patogus, kadangi dauguma naudotojų naudoja mobiliuosius įrenginius;
– Metodą yra santykinai pigu realizuoti;
– Autentifikavimo pranešimai yra siunčiami realiu laiku;
– *Push* pranešimai nėra matomi tol, kol naudotojas neatrakina įrenginio.

Push authentication paremto autentifikavimo metodo trūkumai:

– Jei mobilusis įrenginys yra pavogiamas, naudotojas turi kuo greičiau atšaukti paskyros susiejimą su pavogtu įrenginiu;
– Mobiliajam įrenginiui reikalingas nuolatinis mobiliojo interneto ryšys;
– Naudotojai gali netyčia ar neapdairiai patvirtinti neteisėtas prisijungimo užklausas;
– Prisijungimo užklausa gali būti perimama ir automatiškai patvirtinama telefone veikiančios žalingos *trojan* programinės įrangos;
– Naudotojai turi įsidiegti dedikuotą mobiliąją programą.

1.4.5. Lokacija

Naudotojo lokacija taip pat gali būti panaudota autentifikacijos proceso metu. [28] šaltinyje yra apibūdinamas autentifikacijos procesas naudojant susietą mobiliojo ryšio įrenginį. Autentifikacijos metu yra tikrinama susieto mobiliojo ryšio įrenginio lokacija. Įrenginio lokacija gali būti nustatoma naudojant belaidžio ryšio technologijas, tokias kaip *Wi-Fi* ar *GPS*, bei mobilųjį ryšį – *3G*, *4G* ar *5G*.

Sistemos, kuri autentifikacijos proceso metu atsižvelgia į įrenginio lokaciją, naudojimas dažniausiai susideda iš dviejų procesų: galimų autentifikacijos lokacijų registracijos ir prisijungimo.

Autentifikacijos lokacijų registracijos procesas yra sudarytas iš šių žingsnių:

– Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
– Naudotojas nustatymų lange pateikiamame žemėlapyje nurodo lokacijas, iš kurių yra galimas prisijungimas;
– Lokacijų sąrašas yra išsaugomas ir naudotojui yra aktyvuojama dviejų faktorių autentifikacija.

Prisijungimo procesas susideda iš šių žingsnių:

– Naudotojas, naudodamas mobiliojo ryšio įrenginį, prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
– Sistemai yra siunčiama naudotojo lokacija, pasinaudojant vienu iš lokacijos nustatymo būdu - *Wi-Fi*, mobiliuoju ryšiu ar *GPS*;
– Jei mobiliojo ryšio įrenginio lokacija yra įtraukta į leidžiamų lokacijų sąrašą, naudotojas yra sėkmingai autentifikuojamas ir prijungiamas prie sistemos.

Naudotojo lokacija paremto autentifikavimo metodo privalumai:

– Naudotojui nereikia atlikti papildomų veiksmų, lokacija yra patikrinama automatiškai;

– Suteikiama apsauga nuo bandymų prisijungti iš kitų neleistinų lokacijų - šalių ar kontinentų.

Naudotojo lokacija paremto autentifikavimo metodo trūkumai:

– Lokacija yra privati informacija, kurią ne kiekvienas naudotojas nori atskleisti;
– Lokacijos informacija gali būti klastojama keliuose lygmenyse – techninės įrangos, operacinės sistemos ar programos lygmenyse [29].

1.4.6. Paveikslėlių atpažinimas

[30] šaltinyje yra nurodomas dviejų faktorių autentifikacijos metodas, kuris remiasi susietų paveikslėlių atpažinimu. Autentifikacijos proceso metu naudotojui yra pateikiami keletas paveikslėlių ir naudotojas turi pasirinkti paveikslėlius, kuriuos buvo pasirinkęs paskyros konfigūracijos metu.

Sistemos, kuri autentifikacijos proceso metu naudoja paveikslėlių atpažinimą, naudojimas dažniausiai susideda iš dviejų procesų: susietų paveikslėlių registracijos ir prisijungimo.

Susietų paveikslėlių registracijos procesas yra sudarytas iš šių žingsnių:

– Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
– Naudotojas paskyros nustatymuose iš duotųjų paveikslėlių sąrašo išsirenka vieną ar kelis paveikslėlius, kurie bus naudojami prisijungimo metu;
– Išsaugojus pasirinkimus, naudotojui yra įgalinama dviejų faktorių autentifikacija.

Prisijungimo procesas susideda iš šių žingsnių:

– Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
– Naudotojui yra pateikiamas atsitiktinių paveikslėlių sąrašas, kuriame yra vienas ar keli naudotojo pasirinkti paveikslėliai;
– Naudotojas pasirenka savo paveikslėlius;
– Jei parinkti paveikslėliai yra teisingi, naudotojas yra sėkmingai autentifikuojamas ir prijungiamas prie sistemos.

Paveikslėlių atpažinimu paremto autentifikavimo metodo privalumai:

– Išnaudojamas atpažinimo, o ne prisiminimo žmogaus atminties mechanizmas;
– Sudėtingesnis paveikslėlių dalijimasis tarp kelių asmenų, siekiant dalintis bendra sistemine paskyra;
– Brutalios jėgos atakos tampa sudėtingesnės.

Paveikslėlių atpažinimu paremto autentifikavimo metodo trūkumai:

– Naudotojai užtrunka tam tikrą laiko tarpą rinkdamiesi paveikslėlius iš potencialiai ilgo sąrašo;
– Egzistuoja tikimybė atrinkti paveikslėlius pagal asmens pomėgius bei psichologinę charakteristiką.

1.4.7. *U2F*

Universalus antrasis faktorius (angl. *Universal 2nd Factor*) yra atviras standartas, kuris supaprastina dviejų faktorių autentifikaciją naudojant specializuotus *USB* įrenginius (12 pav. yra pateikiamas įrenginio pavyzdys) ar artimos komunikacijos *NFC* įrenginius ir remiantis išmaniųjų

kortelių saugumo technologijomis. *U2F* standartas buvo sukurtas *FIDO* aljanso, kurį sudaro tokios korporacijos kaip *Google* ir *Microsoft*. *U2F* standartas identiteto patvirtinimui naudoja viešojo rakto kriptografiją, tad privatusis raktas visada yra naudojamas tik privataus asmens.

Šiuo metu (2021-03-19) *U2F* protokolą palaiko *Google Chrome*, *Opera* ir *Firefox* naršyklės. *Internet Explorer* ir *Edge* naršyklės šiuo metu šio protokolo nepalaiko.

Sistemos, kuri autentifikacijos proceso metu naudoja *U2F* protokolą, naudojimas dažniausiai susideda iš dviejų procesų: įrenginio susiejimo ir prisijungimo.

Įrenginio susiejimo procesas susideda iš šių žingsnių:

- Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
- Naudotojas susieja *U2F* įrenginį su paskyra. Jei yra naudojamas *USB* įrenginys, jis yra įkišamas į *USB* prievadą. Įkišus įrenginį, patvirtinimas yra atliekamas paspaudžiant ant įrenginio esantį mygtuką. Susiejimo metu yra generuojami kriptografiniai raktai: privatus raktas lieka įrenginyje, o viešasis yra perduodamas sistemos autentifikacijos serveriui;
- Įrenginys yra susiejamas su paskyra ir naudotojui yra aktyvuojama dviejų faktorių autentifikacija.

Prisijungimo procesas susideda iš šių žingsnių:

- Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
- Naudotojo yra prašoma pateikti susietą *U2F* įrenginį;
- Naudotojas įkiša susietą įrenginį į *USB* prievadą ir patvirtina prisijungimą paspausdamas ant įrenginio esantį mygtuką;
- Įrenginio patvirtinimui yra naudojami kriptografiniai metodai, susiejantys ir patvirtinantys privatų ir viešąjį raktus;
- Jei įrenginys yra patvirtinamas, naudotojas yra sėkmingai autentifikuojamas ir prijungiamas prie sistemos.

12 pav. *YubiKey 5 NFC* įrenginys [31]

U2F protokolu paremto autentifikavimo metodo privalumai:

- Paprasta naudoti;
- Vienas įrenginys gali būti naudojamas autentifikuojantis daugelyje skirtingų sistemų;
- *U2F* raktas yra fizinis įrenginys, kurį padirbti yra sunku;
- Naudojant *U2F* protokolą negalimos perėmimo ir apsimetinėjimo atakos, kadangi privatus raktas visą laiką yra saugomas įrenginyje.

U2F protokolu paremto autentifikavimo metodo trūkumai:

- Tai yra nauja technologija, kuri dar nėra plačiai naudojama ir taikoma;
- Ribotas naršyklių palaikymas;
- *U2F* įrenginiai yra santykinai brangūs.

1.4.8. Biometriniai duomenys

Autentifikacijos proceso metu gali būti naudojami asmens biometriniai duomenys - balsas, veidas, akies rainelė, rankos geometrija, pirštų antspaudai. Biometriniai duomenys yra taikomi autentifikacijos procese, kadangi tam tikri biometriniai duomenys kiekvienam asmeniui yra unikalūs ir juos panaudojus galima unikaliai identifikuoti ir autentifikuoti asmenį.

Biometriniai duomenys gali būti nuskaitomi naudojant įvairius įrenginius: vaizdo kameras, fotoaparatus, pirštų antspaudų skaitytuvus, akies rainelės skaitytuvus ir t.t. Kiekvienai biometrinei informacijai yra naudojami skirtingi įvesties įrenginiai, kurių kaina skiriasi ir yra nemaža.

Sistemos, kuri autentifikacijos proceso metu naudoja biometrinius duomenis, naudojimas dažniausiai susideda iš dviejų procesų: biometrinių duomenų registravimo ir prisijungimo.

Biometrinių duomenų registracijos procesas yra sudarytas iš šių žingsnių:

- Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
- Naudotojas parenka kokie biometriniai duomenys bus naudojami autentifikacijos metu;
- Naudotojas, naudodamas atitinkamą biometrinių duomenų įvesties įrenginį, pateikia savo biometrinę informaciją, pvz. piršto antspaudą;
- Išsaugojus biometrinę informaciją, kuri yra susiejama su naudotojo paskyra, naudotojui yra įgalinama dviejų faktorių autentifikacija.

Prisijungimo procesas susideda iš šių žingsnių:

- Naudotojas prisijungia prie sistemos naudodamas paskyros identifikatorių ir slaptažodį;
- Naudotojas pateikia savo biometrinius duomenis, naudodamas atitinkamą biometrinių duomenų įvesties įrenginį, pvz. pirštų antspaudų skaitytuvą;
- Jei pateikti biometriniai duomenys atitinka registracijos metu pateiktus duomenis, naudotojas yra sėkmingai autentifikuojamas bei prijungiamas prie sistemos.

Biometriniais duomenimis paremto autentifikavimo metodo privalumai:

- Padirbti biometrinius duomenis ir apsimesti kitu asmeniu yra sudėtinga;
- Biometriniai duomenys yra unikalūs ir gali būti naudojami ne tik naudotojo autentifikacijai, bet ir identifikacijai.

Biometriniais duomenimis paremto autentifikavimo metodo trūkumai:

- Biometrinių duomenų naudojimas yra apsaugotas įstatymų;
- Metodo realizavimas yra pakankamai sudėtingas, brangus ir reikalauja papildomos techninės įrangos;

– Sėkmingai padirbus biometrinius duomenis ir apsimetus kitu asmeniu, sukompromitavimas išlieka visam laikui, kadangi teisėtas naudotojas negali pakeisti savo biometrinės informacijos;
– Metodo tikslumas nėra šimtaprocentinis – galimas klaidingas autentifikacijos atmetimas ar patvirtinimas.

1.5. Autentifikacijos paslaugas teikiančių produktų analizė

Šiame poskyryje bus apžvelgiamos kelių faktorių autentifikaciją teikiančios paslaugos ir produktai.

1.5.1. *Auth0*

Ši kompanija teikia identiteto valdymo platformą, kuri apima vieningą prisijungimą, naudotojų paskyrų valdymą bei autentifikaciją naudojant kelis faktorius [32].

Auth0 palaiko naudotojų autentifikavimą naudojant *SMS* žinutes, el. paštą, vienkartinius slaptažodžius (*OTP*) bei *Guardian* mobiliąją programą, kuri realizuoja *push authentication* mechanizmą.

Guardian mobilioji programa autentifikacijos užklausoje pateikia sistemos, prie kurios norima prisijungti, vardą, naudotojo vardą, naršyklės bei operacinės sistemos informaciją, naudotojo lokaciją bei užklausos datą ir laiką. Mobilioji programa veikia su įrenginiais, naudojančiais *Android* ar *iOS* operacines sistemas.

1.5.2. *Okta*

Ši kompanija taip pat teikia identiteto valdymo platformą, kuri apima vieningą prisijungimą, naudotojų paskyrų valdymą bei autentifikaciją naudojant kelis faktorius.

Okta teikia adaptyvią kelių faktorių autentifikaciją [33], t.y. autentifikacijos metu antrojo faktoriaus naudojimas yra nusakomas kontekstinėmis taisyklėmis. Šios taisyklės gali atsižvelgti į vietovės kontekstą (nauja vietovė, neįmanomi keliavimo būdai), įrenginio kontekstą (žinomo įrenginio atpažinimas, įrenginių valdymas) bei tinklo kontekstą (naujas *IP* adresas, specifinės *IP* adresų zonos).

Okta palaiko daug faktorių, kurie gali būti naudojami autentifikacijos metu:

– El. paštas;
– *SMS* žinutės;
– Balso atpažinimas;
– Saugumo klausimai;
– Vienkartiniai slaptažodžiai (*OTP*);
– *FaceID, TouchID, Android Biometrics*;
– *Okta Verify Push* mobilioji programa.

Okta Verify Push mobilioji programa taip pat realizuoja *push authentication* mechanizmą ir užklausos metu pateikia sistemos, prie kurios norima prisijungti, vardą, naudotojo el. pašto adresą, užklausos datą ir laiką bei prisijungimo lokaciją ir *IP* adresą.

Okta taip pat teikia realaus laiko valdymo skydą, kuriame yra matomi platformos įvykiai, pvz. naudotojų prisijungimai ir su jais susietos lokacijos.

1.5.3. *Unloq*

Unloq teikia kelių faktorių autentifikavimo paslaugą [34] naudojant el. paštą, vienkartinius slaptažodžius (*OTP*) ar mobiliąją programą.

Mobilioji programa realizuoja *push authentication* mechanizmą, veikia *Android* bei *iOS* operacinėse sistemose. Mobilioji programa gali būti personalizuojama pagal sistemą, prie kurios norima prisijungti - galima keisti programoje naudojamą logotipą bei spalvas.

Unloq taip pat palaiko veiksmų/transakcijų autorizavimą, t.y. tam tikri naudotojų atliekami veiksmai gali būti patvirtinami naudojant kelių faktorių autentifikavimą, kuomet naudotojas turi patvirtinti veiksmą pasitelkdamas mobiliojo ryšio įrenginį. Tačiau sistema nepalaiko naudotojo prisijungimo prie sistemos proceso autorizavimo.

1.5.4. *Duo*

Duo kompanija taip pat teikia vieningo prisijungimo bei kelių faktorių autentifikacijos paslaugas [35].

Adaptyvios autentifikacijos mechanizmas bei nustatytų taisyklių vykdymas užtikrina, jog neteisėti bandymai prisijungti prie sistemos bus užblokuoti. Taisyklėse galima nurodyti, kokie naudotojai ar naudotojų grupės gali gauti prieigą prie tam tikrų paslaugų, nustatyti reikalavimus geografinėms zonoms ar kompiuterių tinklams, iš kurių galima vykdyti autentifikaciją.

Autentifikavimo proceso metu yra taikomi šie faktoriai:

- *Duo Push* mobilioji programa;
- *SMS* žinutės;
- Mobilieji skambučiai;
- Vienkartiniai slaptažodžiai (*OTP*);
- *U2F* protokolą realizuojantys įrenginiai;
- Žetonų generatoriai.

1.5.5. *RSA SECURID®*

RSA kompanija teikia *SecurID® Suite* paslaugų rinkinį [36], kuris apima naudotojų identiteto valdymą bei autentifikavimo paslaugas.

Identiteto valdymo paslauga teikia automatizuotą paskyrų sukūrimo, priežiūros ir likvidavimo funkcionalumą, rolių valdymą bei išsamias identiteto valdymo ataskaitas.

Kelių faktorių autentifikacija yra vykdoma naudojant žetonų generatorių (programinį ar fizinį), *SecurID® Access* mobiliąją programą, kuri realizuoja *push authentication* mechanizmą, *SMS* žinutes ar biometrinius duomenis, kurie yra gaunami naudojant *TouchID, FaceID, Android fingerprint* ar *Windows Hello* technologijas. Mobilioji programa veikia *Android, iOS* bei *Windows Phone* operacinėse sistemose.

RSA kompanija taip pat siūlo rizika paremtą autentifikacijos metodą, kuris pritaiko kelių faktorių autentifikaciją atsižvelgiant į autentifikacijos užklausos kontekstą – lokaciją, programos jautrumą, sesijos bei tinklo informaciją ar įrenginio tipą.

1.5.6. Paslaugų palyginimas

Apibendrinant autentifikacijos paslaugų ir produktų analizę, 1 lentelėje yra pateikiamas paslaugų palyginimas, atsižvelgiant į jų teikiamas funkcijas bei naudojamus autentifikavimo metodus.

1 lentelė. Kelių faktorių autentifikaciją teikiančių paslaugų palyginimas

Palaikomas funkcionalumas	***Auth0***	***Okta***	***Unloq***	***Duo***	***RSA SECURID®***
SMS žinutės	✔	✔	✘	✔	✔
El. paštas	✔	✔	✔	✘	✘
Vienkartiniai slaptažodžiai/žetonai	✔	✔	✔	✔	✔
Mobilioji programa	✔	✔	✔	✔	✔
Naudotojo lokacija	✘	✔	✘	✘	✔
U2F protokolas	✘	✘	✘	✔	✘
Biometriniai duomenys	✘	✔	✘	✘	✔
Adaptyvi autentifikacija	✘	✔	✘	✔	✔
Autentifikacijos užklausų autorizavimas	✘	✘	✘	✘	✘

1.6. Kritinių infrastruktūrų sistemų autentifikacijos proceso saugos analizė

Kritinių infrastruktūrų sistemos privalo turėti saugos politiką, kuri apimtų skirtingus saugos aspektus bei pateiktų taisykles ir gaires, kuriomis remiantis būtų užtikrinamas sistemos saugumas ir patikimumas. Sudarinėjant saugos politiką, yra svarbu įvertinti rizikas, kylančias šioms sistemoms. [5] šaltinyje yra nurodomos grėsmės, kylančios kritinės infrastruktūros sistemoms, ir šių grėsmių šaltiniai. Literatūroje nurodyti grėsmių šaltiniai yra pateikti 1 priede. Atsižvelgus į grėsmių šaltinius, didžiausią grėsmę sistemoms kelia nepatikimi darbuotojai, valstybių finansuojami nusikaltėliai, įsilaužėliai mėgėjai bei teroristai.

2 priede yra pateikiamas rizikų, gresiančių kritinėms sistemoms, įvertinimo ir šalinimo planas. Lentelėje matoma, jog didžiausios saugos grėsmės, kylančios kritinėms sistemoms, yra neteisėta prieiga prie valdymo sistemų naudojant viešus tinklus (R1) ir nuotolinė prieiga naudojant neapsaugotus vartus (R13). Šios grėsmės atsiranda tuomet, kai kritinės sistemos yra prijungiamos prie viešųjų tinklų ir atsiranda galimybė gauti nuotolinę prieiga prie vidinių valdymo ar priežiūros sistemų.

Saugos rizikos taip pat turi būti suderintos su komercinės veiklos reikalavimais. Reikalavimai yra nustatomi pagal sektorių, normines paklausas, verslo ir finansinius apribojimus ir t.t. Vienas iš dažniausių reikalavimų, skirtų kritinės infrastruktūros sistemoms, yra jų nenutrūkstamas veikimas

24/7/365. Šių sistemų prastova gali turėti ne tik didelių finansinių, bet ir globalių pasekmių. Pavyzdiniai komerciniai reikalavimai, skirti kritinės infrastruktūros sistemoms, yra pateikti 3 priede.

Visgi šie reikalavimai yra pernelyg abstraktūs ir tiesiogiai sunkiai įgyvendinami. Siekiant didesnio konkretumo, yra įvedami komerciniai tikslai, kurie gali apimti ar suskaidyti sudėtingus komercinius reikalavimus ir suteikti galimybę juos valdyti efektyviau. Pavyzdžiui, vienas iš komercinių tikslų gali būti „Suteikti nuotolinę prieigą prie kritinių *SCADA* sistemų tik autentifikuotam priežiūros personalui“, kuris apimtų BR2 ir BR4 reikalavimus. Šio tikslo įtraukimas į saugos politiką užtikrintų, jog aukšto lygio rizikos (R1 ir R13) bus įvertintos ir bus pateikiamos priemonės bei metodai, skirti šių rizikų sumažinimui.

Atsižvelgus į šį iškeltą tikslą, tampa aišku, jog norint užtikrinti, jog kritinė sistema bus saugi ir veiks patikimai, reikalingas saugus sistemų naudotojų autentifikacijos bei autorizacijos mechanizmas, kuris teisingai teiktų nuotolinę prieigą prie kritinių sistemų. Norint pasiekti šį tikslą, reikia įgyvendinti techninius saugos reikalavimus bei taikyti atitinkamas saugumo priemones.

Techniniai standartai pateikia tam tikras priemones bei gaires, kurie leistų įgyvendinti saugos reikalavimus. Remiantis [3] šaltiniu, 13 pav. yra pateikiami organizaciniai ir operaciniai standartai, skirti kritinėms valdymo sistemoms.

	Organizaciniai ir operaciniai standartai							
	NIST 800-53	NISTIR 7628	ISA 99-1	ISA 99-2	ISO 177799	ISO 27001	ISO 27002	ISO 19791
Saugos kontrolė								
Organizacinės saugos pakontrolės								
Saugos politika	✓	✓	✓	✓	✓	✓	✓	
Personalo sauga	✓	✓		✓	✓	✓	✓	✓
Fizinė ir aplinkos sauga	✓	✓	✓	✓	✓	✓	✓	✓
Strateginis planavimas	✓	✓	✓	✓	✓	✓	✓	✓
Saugos suvokimas ir mokymai	✓	✓		✓	✓	✓		✓
Saugos politikos stebėjimas ir peržiūra	✓	✓	✓	✓	✓	✓	✓	
Rizikų valdymas ir įvertinimas	✓	✓	✓	✓	✓	✓	✓	
Saugos programos valdymas	✓	✓						
Operacinės saugos pakontrolės								
Sistemų ir paslaugų įsigijimas	✓	✓	✓	✓	✓	✓		✓
Konfigūracijos valdymas	✓	✓	✓	✓	✓	✓		✓
Sistemų ir komunikacijų apsauga	✓	✓	✓	✓	✓	✓	✓	✓
Informacijos ir dokumentų valdymas	✓	✓		✓	✓	✓		✓
Sistemų kūrimas ir priežiūra	✓	✓	✓	✓	✓	✓	✓	✓
Incidentų valdymas ir atsakas	✓	✓		✓	✓	✓	✓	✓
Sistemų ir informacijos vientisumas	✓	✓			✓	✓		✓
Prieigos kontrolė	✓	✓		✓	✓	✓	✓	✓
Auditas ir atskaitomumas	✓	✓		✓	✓	✓	✓	✓
Informacinės terpės apsauga	✓	✓			✓	✓		

13 pav. Organizaciniai bei operaciniai standartai, skirti kritinėms valdymo sistemoms [3]

Paveikslėlyje matoma, jog beveik visi standartai apžvelgia prieigos kontrolės, į kurią įeina tiek autentifikacijos, tiek autorizacijos procesai, saugos aspektus. Didžioji dalis standartų atsižvelgia į sistemų konfigūracijos valdymo aspektus ir visi standartai apžvelgia sistemų kūrimo ir priežiūros aspektus. Dalis šių standartų bus apžvelgti šiame poskyryje, fokusuojantis į nuotolinės prieigos bei autentifikacijos proceso saugos reikalavimus.

NIST organizacija yra išleidusi specialų karkasą – dokumentą, kuriame yra pateikiamos saugos rekomendacijos kritinės infrastruktūros sistemoms [37]. Karkasas yra sudarytas iš funkcijų,

kategorijų ir pakategorių, kurias reikia įvertinti, siekiant užtikrinti šių sistemų saugą. Karkase yra pateikiama *PR.AC-3: Remote access is managed* pakategorė, kuri patenka į *Identity Management, Authentication and Access Control (PR.AC)* kategoriją ir *Protect (PR)* funkciją. Informacija apie šią pakategorę yra pateikiama 14 paveikslėlyje.

APSAUGOTI (PR)	**Identiteto valdymas, autentifikacija ir prieigos kontrolė (PR.AC):** Prieiga prie fizinių ir loginių išteklių ir susijusios infrastruktūros yra apribota ir pasiekiama tik autorizuotiems naudotojams, procesams ir įrenginiams ir yra nuosekliai valdoma atsižvelgiant į įvertintas rizikas, kuomet yra gaunama neautorizuota prieiga prie autorizuotų veiksmų ir transakcijų.	**PR.AC-1:** Tapatybės ir įgaliojimai yra išduodami, valdomi, tikrinami, atšaukiami ir audituojami, kuomet jais naudojasi autorizuoti įrenginiai, naudotojai ir procesai	**CIS CSC** 1, 5, 15, 16 **COBIT 5** DSS05.04, DSS06.03 **ISA 62443-2-1:2009** 4.3.3.5.1 **ISA 62443-3-3:2013** SR 1.1, SR 1.2, SR 1.3, SR 1.4, SR 1.5, SR 1.7, SR 1.8, SR 1.9 **ISO/IEC 27001:2013** A.9.2.1, A.9.2.2, A.9.2.3, A.9.2.4, A.9.2.6, A.9.3.1, A.9.4.2, A.9.4.3 **NIST SP 800-53 Rev. 4** AC-1, AC-2, IA-1, IA-2, IA-3, IA-4, IA-5, IA-6, IA-7, IA-8, IA-9, IA-10, IA-11
		PR.AC-2: Fizinė prieiga prie išteklių yra valdoma ir apsaugota	**COBIT 5** DSS01.04, DSS05.05 **ISA 62443-2-1:2009** 4.3.3.3.2, 4.3.3.3.8 **ISO/IEC 27001:2013** A.11.1.1, A.11.1.2, A.11.1.3, A.11.1.4, A.11.1.5, A.11.1.6, A.11.2.1, A.11.2.3, A.11.2.5, A.11.2.6, A.11.2.7, A.11.2.8 **NIST SP 800-53 Rev. 4** PE-2, PE-3, PE-4, PE-5, PE-6, PE-8
		PR.AC-3: Nuotolinė prieiga yra valdoma	**CIS CSC** 12 **COBIT 5** APO13.01, DSS01.04, DSS05.03 **ISA 62443-2-1:2009** 4.3.3.6.6 **ISA 62443-3-3:2013** SR 1.13, SR 2.6 **ISO/IEC 27001:2013** A.6.2.1, A.6.2.2, A.11.2.6, A.13.1.1, A.13.2.1

14 pav. *Identity Management, Authentication and Access Control* saugos reikalavimų kategorija [37]

Prie šios pakategorės yra pateikiamas sąrašas susijusių dokumentų ir standartų, kuriuose yra pateikiamos gairės, skirtos užtikrinti šios pakategorės saugos reikalavimus. [3] ir [37] šaltiniuose yra minimas *NIST SP 800-53* dokumentas. Tai yra *NIST* organizacijos išleistas dokumentas pavadinimu *Security and Privacy Controls for Federal Information Systems and Organizations* [38], kuriame yra apžvelgiamos saugumo ir privatumo priemonės, taikomos federalinėse informacinėse sistemose ir organizacijose. Viena iš saugumo priemonių, apžvelgiamų šiame dokumente yra *IA-2 Identification And Authentication (Organizational Users)*. Šioje priemonėje yra minima, jog yra svarbu teisingai identifikuoti ir autentifikuoti nuotolinius sistemos naudotojus. Naudotojai gali būti autentifikuojami naudojant standartines priemones – slaptažodžius – arba taikant kelių faktorių autentifikavimo metodus įtraukiant ir skaitmeninių sertifikatų panaudojimą. Nuotolinė prieiga taip pat turi būti apsaugota naudojant *VPN* priemones. Šiame dokumente yra pristatoma *out-of-band authentification* sąvoka ir reikalavimas, kuris nurodo, jog taikant kelių faktorių autentifikaciją, turi būti naudojami keli skirtingi kanalai – skirtingi fiziniai įrenginiai, kuriais autentifikacijos metu yra pateikiama informacija.

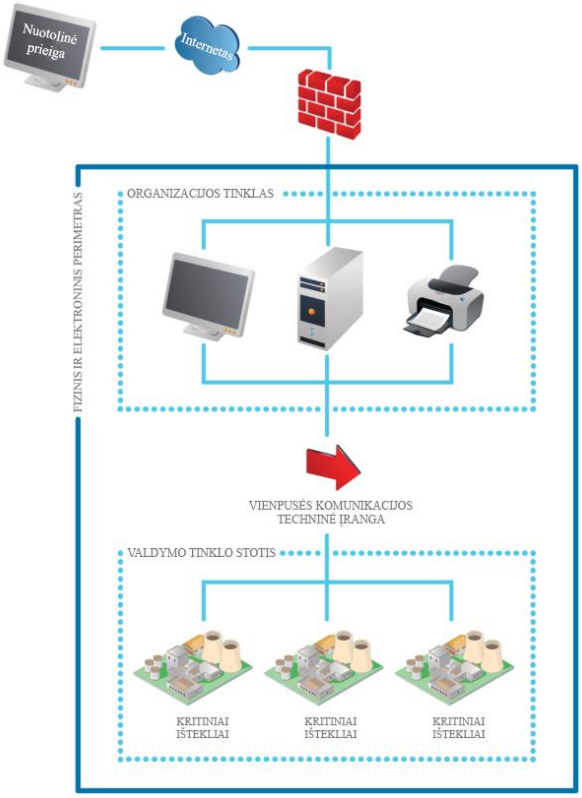
[39] šaltinio 5.10.2 skyriuje yra nurodomi saugos reikalavimai, skirti industrinių valdymo sistemų, kurios taip pat patenka į kritinės infrastruktūros kategoriją, nuotolinei prieigai prie ugniasienių valdymo skydo. Reikalavimuose nurodoma, jog nuotolinei prieigai gauti, reikalinga naudoti *VPN* klientą, apsaugoti komunikaciją *HTTPS* protokolu bei naudoti stiprią kelių faktorių autentifikaciją. Taip pat nurodoma, jog tarpiniai (angl. *proxy*) serveriai gali būti taikomi siekiant užtikrinti nuotolinės prieigos apsaugą. Šie saugos reikalavimai yra taikomi ugniasienių valdymui, tačiau taip pat gali būti pritaikyti ir kitiems kritinių infrastruktūrų sistemų valdymo komponentams, pvz. klasterio valdymo skydui, kadangi klasterio valdymo funkcijos yra ne mažiau svarbios bei būtinos nei ugniasienės, siekiant užtikrinti sėkmingą ir saugų kritinių sistemų veikimą.

[40] ir [41] šaltiniuose yra apžvelgiami reikalavimai, kurie yra nurodyti *CIS* (angl. *Center for Internet Security*) organizacijos sudarytame dokumente [42], nurodančiame kritinių infrastruktūrų

sistemų saugos reikalavimus. Dvyliktasis reikalavimas yra ribinė apsauga. Šiame reikalavime yra nurodoma, jog norint gauti nuotolinę prieigą prie vidiniame organizacijos tinkle esančių sistemų, turi būti naudojamas užšifruotas kanalas (*VPN*) bei naudojama kelių faktorių autentifikacija, siekiant saugiai autentifikuoti naudotoją.

NCCIC (angl. *National Cybersecurity and Communications Integration Center*) centro išleistame industrinių valdymo sistemų saugos strategijos dokumente [43] taip pat yra nurodoma, jog yra privaloma užtikrinti, jog nuotolinė prieiga prie tokio tipo sistemų būtų kuo saugesnė ir patikimesnė, realizuojant dviejų faktorių autentifikaciją nutolusių naudotojų autentifikavimui.

Galiausiai, Entrust organizacija išleido leidinį [44], kuriame yra apibendrinami *NERC* (angl. *North American Electric Reliability Corporation*) organizacijos sudaryti *CIP* (angl. *Critical infrastructure protection*) standartai, skirti kritinės infrastruktūros apsaugai. *CIP-003 Security Management Controls* standarte yra nurodoma, jog stipri kelių faktorių autentifikacija turi būti taikoma norint gauti nuotolinę prieigą prie vidiniuose tinkluose esančių kritinių išteklių. 15 pav. yra pateikta leidinyje nurodyta kritinės infrastruktūros tinklo topologija, iliustruojanti nuotolinę prieigą prie vidiniame tinkle esančių kritinių resursų.



15 pav. Aukšto lygio kritinės infrastruktūros tinklo topologija [44]

Visgi nuotolinei prieigai gauti neužtenka vien naudotojų autentifikacijos – reikalinga ir nuotolinės prieigos autorizacija.

Australijos kibernetinio saugumo centras pateikia publikaciją [45], kurioje yra nurodomas protokolas, skirtas rangovų nuotolinei prieigai prie industrinių valdymo sistemų. Pirmiausiai yra minima, jog nuotolinių naudotojų autentifikacijai turi būti naudojamas kelių faktorių autentifikacijos metodas. Antra – turi būti nustatyta procedūra, kuri leistų aukštesnio rango darbuotojui patvirtinti

nuotolinės prieigos prašymą – t.y. autorizuoti nuotolinio naudotojo prieigą. [46] šaltinyje taip pat yra minimas formalus nuotolinės prieigos prie industrinės valdymo sistemos suteikimo procesas, kuomet nuotolinė prieiga yra patvirtinama vyresniojo vadovo. [39] šaltinyje yra minima, jog egzistuoja poreikis turėti nuotolinę prieigą prie industrinių valdymo sistemų šių sistemų priežiūros metu, tad nuotolinė prieiga yra svarbi ne tik naudojant, bet ir prižiūrint tokio tipo sistemas.

[47] šaltinyje yra pateikiamos saugos priemonės nuotolinei prieigai prie išmanių tinklų (angl. *smart grid*) sistemų. SG.AC-2 ir SG.AC-15 reikalavimai nurodo, jog organizacija privalo autorizuoti nuotolinės prieigos prašymus prie vidinių sistemų bei taikyti automatizuotus metodus, stebint bei prižiūrint nuotolinę prieigą. SG.MA-6 reikalavimas apibūdina nuotolinę šių sistemų priežiūrą. Reikalavime nurodoma, jog priežiūros personalas turi pranešti sistemos administratoriui kada bus atliekami priežiūros darbai bei gauti nuotolinės priežiūros patvirtinimą iš vadovybės.

[38] šaltinyje taip pat yra minimi autorizacijos reikalavimai nuotolinei prieigai. AC-17 nuotolinės prieigos reikalavimo „b“ punkte nurodoma, jog organizacija turi autorizuoti nuotolinę prieigą prie informacinės sistemos prieš suteikiant nuotolinę prieigą. MA-4 nevietinės priežiūros reikalavime taip pat yra nurodoma, jog organizacija patvirtina ir stebi nevietinę sistemos priežiūrą bei užtikrina stiprių autentifikatorių (kelių faktorių autentifikacijos) taikymą užmezgant nuotolines sesijas.

Kritinės infrastruktūros sistemose yra labai svarbu užtikrinti teisingus prieigos kontrolės mechanizmus: identifikuoti, autentifikuoti bei autorizuoti naudotojus. Vis dėlto kritinėse sistemose dažnai yra taikomi paprasti autentifikavimo mechanizmai – pateikiant naudotojo vardą ir slaptažodį. Tyrimuose yra nustatyta [48], jog slaptažodžiu paremti autentifikavimo mechanizmai turi daug pažeidžiamumų (žodyno atakos, brutalios jėgos atakos, klaviatūros paspaudimus fiksuojančios programos) ir kelia didelę grėsmę sistemų saugumui. Kelių faktorių autentifikacijos mechanizmo pritaikymas leistų užtikrinti, jog prieigos prie svarbių ir kritinių resursų mechanizmas būtų saugesnis ir leistų apsisaugoti nuo tam tikrų rizikų ir atakų, kuriuos būtų įmanomos naudojant tik vieno faktoriaus autentifikacijos mechanizmą. Taip pat yra svarbu autorizuoti naudotojus ir nurodyti, kur ir kada jiems yra leidžiama atlikti tam tikrus veiksmus su sistema, siekiant apsisaugoti nuo netyčinių ar neteisėtų veiksmų, kurie galėtų sutrikdyti sistemų veiklą.

Remiantis 1.5 poskyryje atliktos autentifikacijos veiksnių bei 1.6 poskyryje autentifikacijos paslaugas teikiančių produktų analizės rezultatais galima teigti, jog šiuo metu egzistuojantys autentifikavimo metodai bei juos taikančios paslaugos netenkina kritinės infrastruktūros sistemoms keliamų autentifikacijos reikalavimų. Autentifikavimo metodų, kurie įgyvendintų kritinės infrastruktūros sistemų keliamus saugos reikalavimus, trūkumas sudaro prielaidas mokslinės problemos formulavimui, t.y. egzistuoja poreikis pasiūlyti naują autentifikacijos metodą, kuris būtų taikomas apsaugant nuotolinę prieigą prie kritinės infrastruktūros objektų ir išpildytų šiuos keliamus saugos reikalavimus:

1. Nuotolinių naudotojų autentifikacijai turi būti naudojamas stiprus kelių faktorių autentifikacijos metodas, kuris remiasi keliais kanalais (angl. *out-of-band authentication*);
2. Turi būti nustatyta procedūra, kuri leistų aukštesnio rango darbuotojui patvirtinti nuotolinės prieigos prašymą;
3. Nuotolinė prieiga bei autentifikacijos procesas turi vykti naudojant saugų informacijos perdavimo kanalą (*SSL* protokolą, naudotojų sertifikatus bei *VPN* tunelius).

Atsižvelgus į šiuos reikalavimus, siūlomas autentifikacijos metodas galėtų naudoti mobiliojo ryšio įrenginį, taip įgyvendinant pirmąjį kelių kanalų panaudojimo reikalavimą. Taip pat siūlomas metodas turėtų suteikti galimybę aukštesnio rango darbuotojui patvirtinti nuotolinės prieigos prašymą. Galiausiai, metodo taikymo metu visi duomenys turi būti perduodami saugiais kanalais, pasitelkiant *SSL* protokolą bei naudotojų skaitmeninius sertifikatus.

1.7. *Kubernetes* konteinerių klasterio ir valdymo skydelio analizė

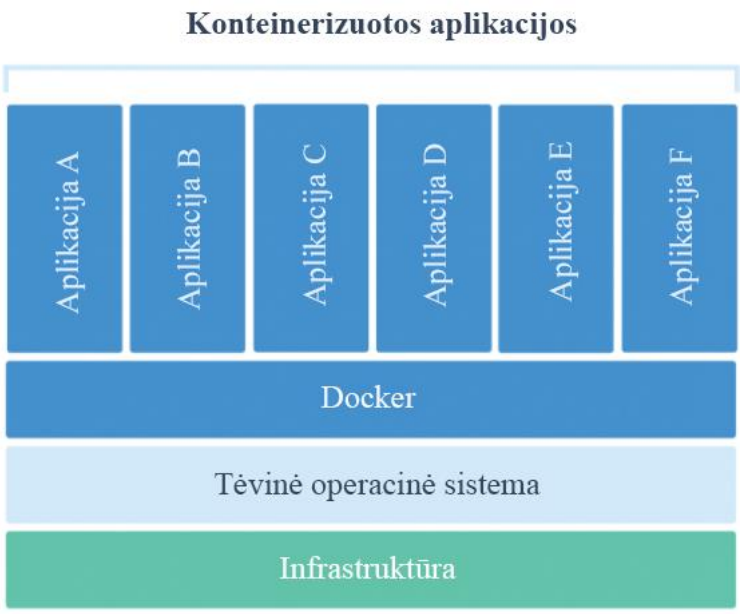
Kritinės infrastruktūros sistemos vis dažniau yra konteinerizuojamos ir diegiamos konteinerių klasteriuose tokiuose kaip *Kubernetes* ar *OpenShift*. [49] straipsnyje rašoma, jog viena iš kritinių *CERN* organizacijos sistemų buvo sėkmingai konteinerizuota bei įdiegta į *OpenShift* konteinerių klasterį.

[50] šaltinyje yra minima, jog kritinės infrastruktūros sistemos vis dažniau yra keliamos į debesijos paslaugas. Kritinės infrastruktūros sistemos yra paskirstomos į atskirus komponentus, kuriems yra taikomi skirtingi atsparumo bei plėtimo reikalavimai. Viena iš migravimo priežasčių yra dinaminis skaičiavimo galios bei duomenų talpyklos plėtimas, kuomet kritinės infrastruktūros sistemos apdoroja didelius kiekius duomenų. Straipsnyje taip pat yra pabrėžiama, jog reikalavimai kritinės infrastruktūros sistemoms yra žymiai griežtesni nei įprastoms komercinėms sistemoms, atsižvelgiant į šiuos aspektus: pertekliškumą, duomenų prieinamumą, autentiškumą, saugią prieigą bei žemą tinklo vėlinimą. Taip pat tokio tipo sistemos kelia aukštesnius reikalavimus saugumui, patikimumui bei atsparumui, kuomet yra kalbama apie debesijos aplinkas.

Keliuose komerciniuose pristatymuose [51] [52] yra pateikiami pavyzdžiai, kaip konteinerizacija bei tokios konteinerių valdymo sistemos kaip *OpenShift* gali būti pritaikytos kritinės infrastruktūros sistemose: kibernetinio karo bei naftos ir dujų gavybos sistemose.

Tad remiantis šaltiniais galima teigti, jog konteinerių klasterių sistemų panaudojimas tampa vis dažnesnis ne tik komercinėse sistemose, bet ir kritinės infrastruktūros sistemose. Tačiau taip pat reikia įvertinti, ar šios sistemos tenkina kritinės infrastruktūros sistemoms keliamus saugos reikalavimus. Norint tai atlikti, pirmiausiai reikia suprasti kaip šios sistemos veikia. Šio darbo metu kaip pavyzdys bus pasirinkta *Kubernetes* konteinerių klasterio sistema ir bus išanalizuoti jos baziniai veikimo, naudojimo bei saugos principai.

Konteineris – tai standartinis programinės įrangos vienetas, kuris yra sudarytas iš programos artefaktų ir programos paleidimui reikalingos programinės įrangos, pvz. programos serverio. Konteinerių sukūrimui dažniausiai yra naudojama *Docker* programinė įranga. Serveryje yra įdiegiamas *Docker* servisas, kurio pagalba gali būti paleidžiami programų konteineriai. 16 pav. yra pavaizduotas konteinerių veikimo lygmuo.

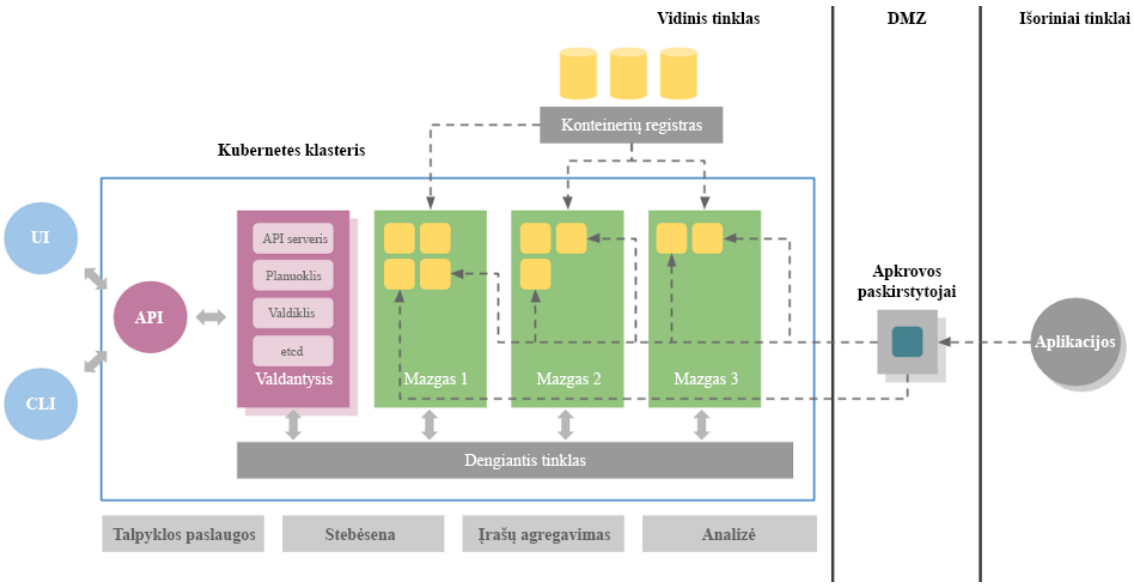


16 pav. *Docker* konteinerių veikimo lygmuo [53]

Visgi konteinerius valdyti serveryje rankiniu būdu yra nepatogu ir nepatikima, tad tam yra naudojamos dedikuotos konteinerių valdymo sistemos. *Kubernetes* – tai atviro kodo sistema, skirta konteinerizuotų programų automatiniam diegimui, plėtimui bei valdymui [54].

Kubernetes sistema pateikia tam tikras abstrakcijas, kurios palengvina konteinerizuotų programų diegimą ir priežiūrą. Servisai suteikia vieningą prieigą prie replikuotų tos pačios programos konteinerių ir leidžia tarp jų padalinti srautą. Diegimų rinkiniuose galima nurodyti kiek aktyvių tos pačios programos konteinerių turi būti paleidžiama, siekiant užtikrinti horizontalų programos plėtimą. Konfigūraciniai žemėlapiai (angl. *ConfigMap*) suteikia galimybę tam tikrą jautrią konfigūracinę informaciją (pvz. duomenų bazės prisijungimo duomenis) perduoti konteineriams.

Kubernetes klasteris yra sudarytas iš keleto skirtingų komponentų, kurie yra pavaizduoti 17 pav.



17 pav. *Kubernetes* klasterio architektūra [55]

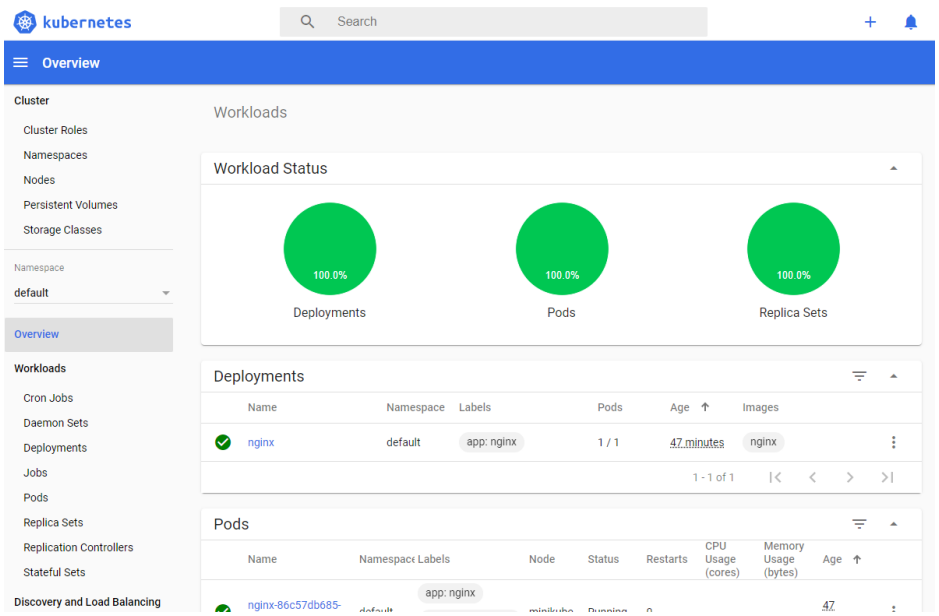
Kubernetes klasteris yra sudarytas iš keleto serverių, vadinamų mazgais (angl. *nodes*). Mazgų serveriuose yra paleidžiami konteineriai. Valdantieji serveriai (angl. *masters*) yra atsakingi už konteinerių paleidimą, sustabdymą ir kitas klasterio valdymo funkcijas.

Konteinerių registre yra saugomi konteinerių atvaizdai (angl. *images*), kurie yra naudojami kaip šablonai kuriant konteinerius. Apkrovos paskirstytojai yra atsakingi už srauto į programas paskirstymą skirtingiems konteineriams. Taip pat egzistuoja pagalbiniai komponentai, skirti klasterio veikimo stebėjimui, sisteminio žurnalo surinkimui, analitinėms funkcijoms bei talpyklų servisams.

Norint klasteryje atlikti tam tikrus administracinius veiksmus, yra naudojamas *API* komponentas, kuris komunikuoja su valdančiaisiais serveriais. *API* yra pasiekiamas naudojant komandinės eilutės įrankį arba grafinę vartotojo sąsają – valdymo skydelį. Naudojant šį *API* galima atlikti įvairius administravimo veiksmus. Dažniausiai yra sukuriami arba modifikuojami įvairūs klasterio objektai – servisai, diegimų rinkiniai, replikų rinkiniai, individualūs konteineriai, konfigūraciniai žemėlapiai ir t.t.

Komandinės eilutės įrankis dažniausiai yra naudojamas nuolatinės integracijos ir diegimo serveriuose ir juose esančiuose diegimo scenarijuose (angl. *CI/CD pipelines*), o grafinė vartotojo sąsaja – valdymo skydelis – yra naudojamas priežiūros personalo.

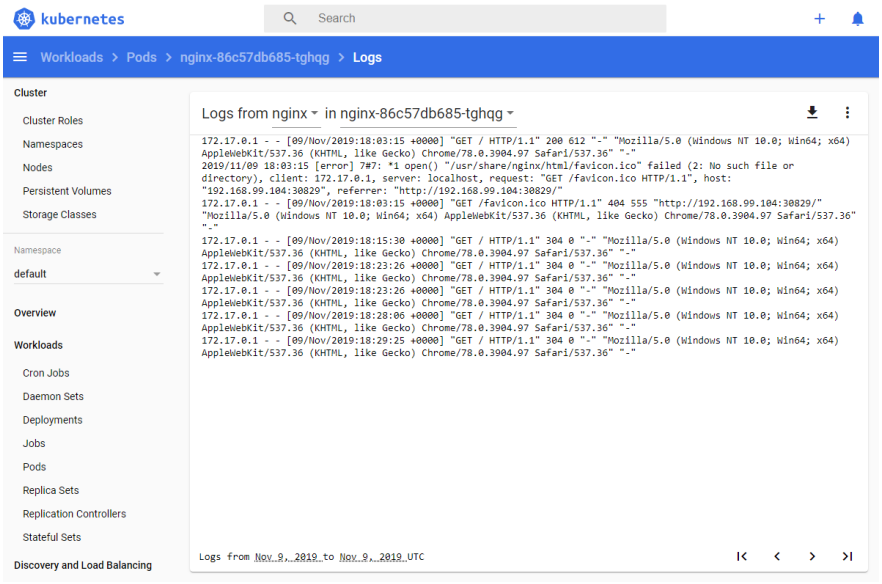
Priežiūros personalas gali pasiekti *Kubernetes* klasterio valdymo skydelį [56], kuriame pateikiama išsami informacija apie klasterio objektus: paslaugas, diegimus, konteinerių rinkinius. 18 pav. yra pateikiamas pavyzdinis *Kubernetes* klasterio valdymo skydelis.



18 pav. *Kubernetes* klasterio valdymo skydelis

Valdymo skydelyje priežiūros personalas gali keisti kiekvienos programos konteinerių skaičių. Taip pat priežiūros personalas gali pašalinti neteisingai veikiančius konteinerius, kurių vietoje automatiškai yra sukuriami nauji.

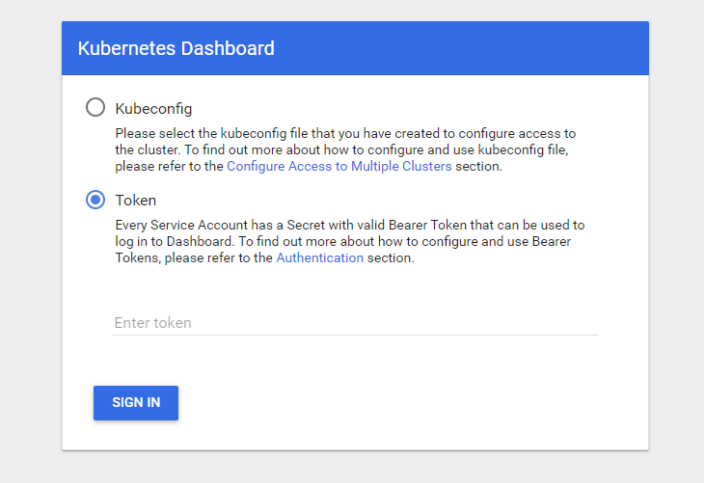
Klasterio valdymo skydelyje taip pat yra pateikiami kiekvieno konteinerio išvedami sisteminiai įrašai. 19 pav. yra pateikti pavyzdiniai konteinerio išvedami sisteminiai įrašai.



19 pav. Konteinerio išvedami sisteminiai įrašai

Valdymo skydelio dokumentacijoje [56] rašoma, jog skydelio prieiga yra kontroliuojama naudojant tik autentifikavimo žetoną ir kiti prieigos apsaugos mechanizmai, pvz. išoriniai identiteto tiekėjai ar sertifikatais grįstas autentifikavimas, nėra palaikomi.

Norint gauti prieigą prie valdymo skydelio, yra reikalingas prieigos žetonas (angl. *Bearer Token*). Tai yra saugos žetonas, pasižymintis savybe, jog bet kas, turintis šį žetoną, gali jį panaudoti, norint gauti prieigą prie resurso ir iš žetono turėtojo neprašomas papildomas žetono nuosavybės patvirtinimas [57]. 20 pav. yra pavaizduotas valdymo skydelio prisijungimo langas.



20 pav. *Kubernetes* valdymo skydelio prisijungimo langas

Prieigos žetonas yra gaunamas sukuriant *ServiceAccount* klasterio resursą ir jam priskiriant klasterio administratoriaus rolę. Tuomet naudojant komandinės eilutės įrankį, yra gaunamas *ServiceAccount* resurso prieigos žetonas, kuris yra naudojamas prisijungiant prie valdymo skydelio.

Visgi tai nėra saugus būdas kontroliuoti prieigą prie valdymo skydelio. Bet kuris asmuo, turėdamas prieigos žetoną, gali prisijungti prie valdymo skydelio ir atlikti įvairius veiksmus, kurie gali sutrikdyti klasterio darbą. Kuomet klasteryje yra diegiamos kritinės infrastruktūros sistemos, jo sutrikimai gali turėti didelių pasekmių kritinės infrastruktūros veiklai.

Kuomet klasterį prižiūri keletas žmonių, atsiranda poreikis kiekvienam asmeniui sukurti dedikuotą *ServiceAccount* resursą arba visam personalui dalintis tuo pačiu žetonu. Naujo resurso kūrimas yra pakankamai nepatogus, kadangi viskas yra atliekama komandinės eilutės įrankio pagalba, o žetono dalijimasis taip pat nėra pati geriausia praktika. Dalijantis tuo pačiu žetonu prarandama galimybė atsekti, kuris asmuo vykdė tam tikrus veiksmus valdymo skydelyje.

Atlikus *Kubernetes* sistemos veikimo ir nuotolinės prieigos prie jos valdymo skydelio saugos analizę, galima teigti, jog nuotolinė prieiga nėra pakankamai saugi, kadangi yra naudojamas tik statinio prieigos žetono mechanizmas. Norint užtikrinti, jog nuotolinė prieiga prie valdymo skydelio būtų saugi ir tenkintų kritinės infrastruktūros sistemų saugos reikalavimus, reikia taikyti papildomas priemones, kurios užtikrintų, jog tik patikimai autentifikuoti bei autorizuoti asmenys galėtų gauti prieigą prie valdymo skydelio funkcionalumo.

Remiantis 1.7 poskyryje atliktos kritinių infrastruktūrų sistemų autentifikacijos proceso saugos analizės rezultatais, galima teigti, jog šiuo metu nėra tinkamo autentifikacijos metodo, kuris realizuotų visus keliamus saugos reikalavimus tokio tipo sistemoms ir užtikrintų saugią nuotolinę prieigą prie *Kubernetes* valdymo skydelio. Tad egzistuoja poreikis pasiūlyti naują autentifikacijos metodą ir jį pritaikyti apsaugant nuotolinę prieigą prie valdymo skydelio.

1.8. Analizės išvados

Analizės metu buvo identifikuota kritinės infrastruktūros sistemų sąvoka, pobūdis bei jų taikymo praktinė reikšmė visuomenės gerovei.

Atlikus sistemų saugos analizę ir nustačius šioms sistemoms kylančias grėsmės bei specifinius nuotolinės prieigos kontrolės saugos reikalavimus, nustatyta, jog kritinės infrastruktūros sistemų nuotoliniam prieigos valdymui turi būti užtikrinta, kad: 1) nuotolinių naudotojų autentifikacijai bus naudojamas stiprus kelių faktorių autentifikacijos metodas, kuris remiasi keliais kanalais (angl. *out-of-band authentication*); 2) yra nustatyta procedūra, kuri leistų aukštesnio rango darbuotojui patvirtinti nuotolinės prieigos prašymą, siekiant išvengti neautorizuotų asmenų nuotolinės prieigos prie kritinės infrastruktūros sistemų; 3) nuotolinė prieiga bei autentifikacijos procesas vyktų naudojant saugų informacijos perdavimo kanalą (*HTTPS* protokolą bei *VPN* tunelius).

Autentifikacijos proceso analizės metu buvo nustatyta, kad dalis autentifikavimo metodų nėra tinkami, kadangi netenkina *out-of-band authentication* reikalavimo ir naudoja tuos pačius informacijos pateikimo kanalus. Vieno tipo kanalo (pvz. tik naudotojo kompiuterio) naudojimas kelių faktorių autentifikacijos mechanizme padidina riziką, jog naudojant tik vieno tipo kanalą, jo pažeidimo metu bus pažeistas visas autentifikacijos procesas. Kelių tipų kanalų panaudojimas (pvz. naudotojo kompiuteris ir mobilusis įrenginys ar fizinis žetonų generatorius) užtikrintų didesnį saugos lygį, kadangi pažeidus tik vieną iš kanalų, kitas kanalas užtikrintų autentifikacijos proceso apsaugą nuo pažeidimo. Taip pat, *SMS* žinučių metodas nebėra laikomas saugiu, o naudotojo lokacijos bei biometrinių duomenų metodai nėra paprastai realizuojami bei pritaikomi. Tinkamiausiu metodu buvo išrinktas mobilaus ryšio įrenginio panaudojimas, tačiau šis metodas taip pat turi būti plečiamas, siekiant įgyvendinti antrąjį kritinės infrastruktūros reikalavimą autentifikacijos procesui – autentifikacijos užklausos patvirtinimą, siekiant užtikrinti, jog tik vadovo autorizuoti asmenys galėtų gauti nuotolinę prieigą prie kritinės infrastruktūros sistemų.

Atlikus egzistuojančių autentifikacijos paslaugas teikiančių produktų analizę, nustatyta, kad nė vienas iš egzistuojančių produktų nesuteikia galimybės vykdyti kelių žingsnių autentifikacijos proceso su galimybe kitam asmeniui patvirtinti autentifikacijos ir prieigos suteikimo užklausą.

Atlikus autentifikacijos metodų bei juos taikančių paslaugų analizę, buvo nustatyta, jog nė vienas iš analizuotų metodų netenkina visų trijų kritinės infrastruktūros sistemų saugos reikalavimų, keliamų autentifikacijos mechanizmui, tad egzistuoja poreikis pasiūlyti naują metodą, kuris realizuotų šiuos reikalavimus.

Konteinerių klasterio valdymo sistemos analizė parodė, kad *Kubernetes* sistemos valdymo skydelio apsauga yra pakankamai paprasta, kadangi projektuojant valdymo skydelį nebuvo numatyta, jog gali egzistuoti poreikis turėti ne tik lokalią prieigą iš klasteriui priklausančio įrenginio, bet ir nuotolinę prieigą prie šio skydelio. To pasekoje, *Kubernetes* sistema neteikia tam tikrų reikiamų saugumo priemonių, skirtų apsaugoti nuotolinę prieigą prie valdymo skydelio, tad norint ją pritaikyti kritinės infrastruktūros sistemoje, reikalinga sukurti bei pritaikyti papildomus saugos mechanizmus, kurie leistų įgyvendinti keliamus saugos reikalavimus. Siekiant įgyvendinti šiuos keliamus reikalavimus, egzistuoja poreikis naują pasiūlytą autentifikacijos metodą pritaikyti saugiai nuotolinei prieigai prie kritinės sistemos.

2. Kelių faktorių autentifikacijos metodo ir jį palaikančios sistemos projektas

Projektinės dalies tikslas - suprojektuoti sistemą, kuri valdytų naudotojų paskyras, realizuotų kelių faktorių autentifikacijos ir autorizacijos metodą bei suteiktų nuotolinę prieigą prie *Kubernetes* valdymo skydelio. Šiam tikslui pasiekti yra keliami šie uždaviniai:

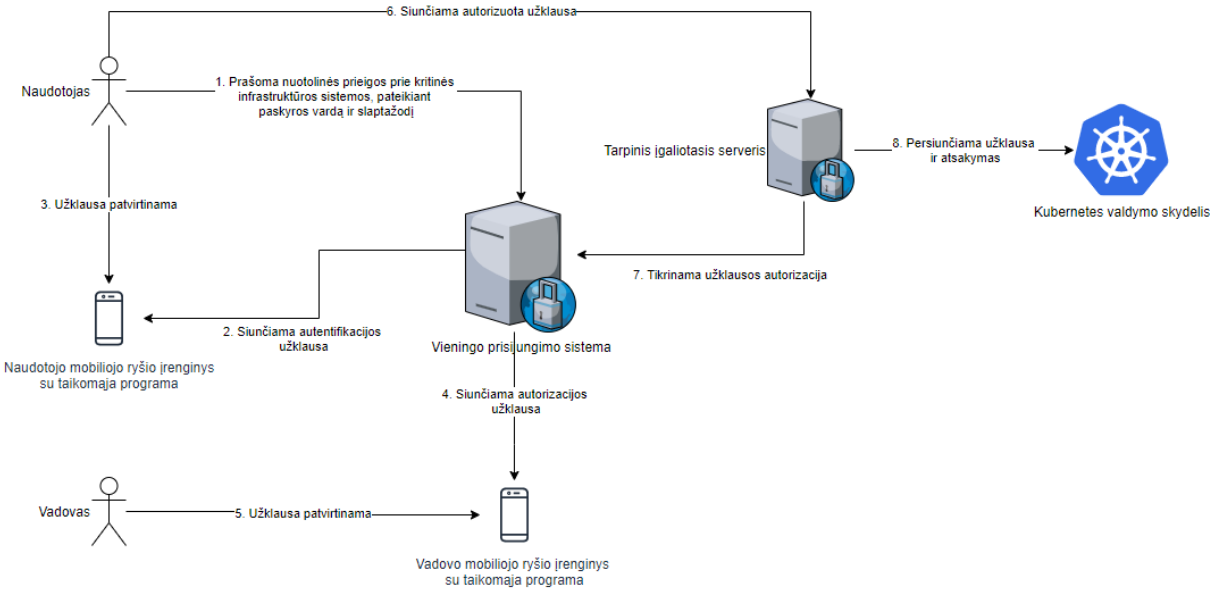
1. Sudaryti problemos sprendimo aukšto lygio koncepciją;
2. Nustatyti sistemos realizacijai keliamus funkcinius ir nefunkcinius reikalavimus;
3. Sudaryti sistemos modelį, pasitelkiant *UML* diagramas:
    a. Diegimo;
    b. Paketų;
    c. Veiklos.

2.1. Sprendimo koncepcija

Siekiant užtikrinti saugią nuotolinę prieigą prie *Kubernetes* valdymo skydelio, sprendime bus taikomos šios saugos priemonės:

1. Išjungta tiesioginė nuotolinė prieiga prie valdymo skydelio;
2. Išjungtas valdymo skydelyje taikomas numatytasis nesaugus autentifikacijos mechanizmas, kuris remiasi statiniais prieigos žetonais;
3. Sukurtas tarpinis įgaliotasis serveris (angl. *proxy*), kuris būtų pasiekiamas iš išorės ir perduotų užklausas į valdymo skydelį ir grąžintų atsakymus iš jo. Šis serveris būtų vienintelis taškas, per kurį vyktų komunikacija su valdymo skydeliu;
4. Sukurta vieningo prisijungimo sistema, kuri būtų atsakinga už naudotojų autentifikacijos ir autorizacijos procesą, taikant stiprų kelių faktorių autentifikacijos metodą;
5. Apjungta vieningo prisijungimo sistema ir tarpinis įgaliotasis serveris, siekiant užtikrinti, jog visos nuotolinės užklausos į tarpinį įgaliotąjį serverį yra autorizuotos ir užklausų autorius yra autentifikuotas. Neautorizuotos ir neautentifikuotos užklausos nebus praleidžiamos į valdymo skydelį;
6. Visos užklausos kelyje nuo naudotojo naršyklės iki valdymo skydelio vyksta apsaugotu kanalu (*HTTPS* protokolu).

21 pav. yra pateiktas tokio sprendimo koncepcinis modelis. Vieningo prisijungimo sistema būtų atsakinga už naudotojų autentifikacijos ir autorizacijos procesą. Tarpinis įgaliotasis serveris valdytų prieigą prie kritinės infrastruktūros sistemos: suteiktų nuotolinę prieigą, jei autentifikacijos/autorizacijos procesai buvo įvykdyti sėkmingai ir užklausos informacija yra teisinga, arba uždraustų nuotolinę prieigą, jei procesai nebuvo įvykdyti sėkmingai ar užklausos informacija yra neteisinga.



21 pav. Sprendimo koncepcinis modelis

Naudotojas, norėdamas gauti nuotolinę prieigą prie sistemos, turėtų sėkmingai įvykdyti trijų žingsnių autentifikacijos/autorizacijos procesą:

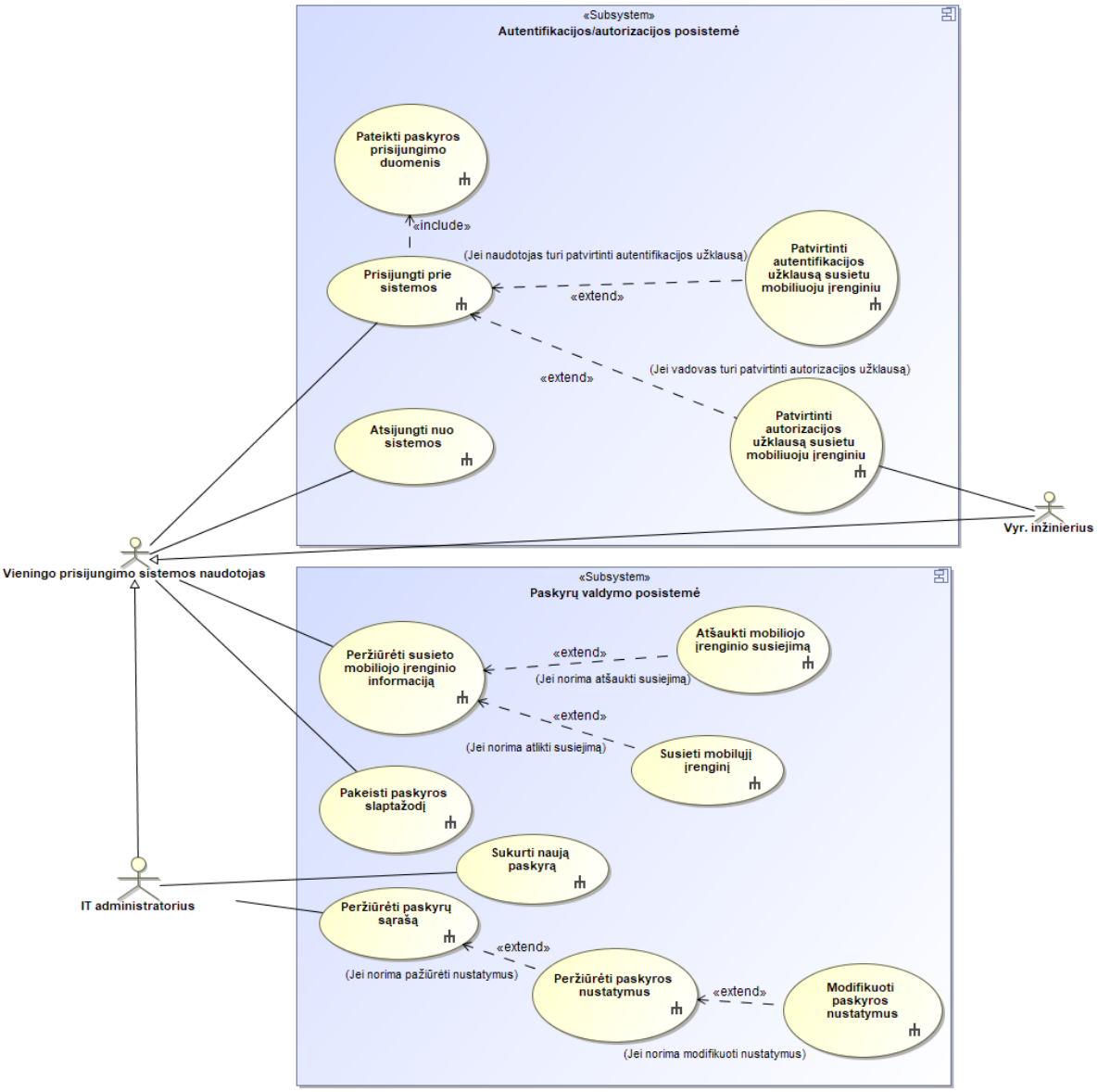
1. Pirmasis žingsnis yra pateikti paskyros identifikatorių ir su juo susietą slaptažodį, kuri naudotojas gauna iš administratoriaus, kuomet šis sukuria paskyrą. Jei duomenys yra teisingi - naudotojui į jo susietą mobiliojo ryšio įrenginį bus nusiunčiama autentifikacijos užklausa;
2. Kuomet naudotojas sėkmingai patvirtina autentifikacijos užklausą, jo vadovams yra nusiunčiama autorizacijos užklausa, kurioje yra klausiama, ar pavaldiniui yra leidžiama jungtis prie tam tikros sistemos ir yra suteikiama nuotolinė prieiga;
3. Kuomet vienas iš nurodytų vadovų sėkmingai patvirtina autorizacijos užklausą, naudotojui yra suteikiama nuotolinė prieiga prie apsaugotos sistemos. Sistema turi suteikti galimybę nurodyti kelis galimus naudotojo vadovus, siekiant padidinti pertekliškumą tais atvejais, kai vieno iš vadovų mobilusis įrenginys nėra pasiekiamas.

Realizavus tokį sprendimą, būtų užtikrinama, jog nuotolinė prieiga prie kritinės infrastruktūros sistemos yra valdoma taikant saugumo mechanizmus, kurie yra pateikiami tokio tipo sistemų saugos reikalavimų dokumentuose.

2.2. Realizacijai keliami reikalavimai

2.2.1. Funkciniai reikalavimai

Sistemos funkciniai reikalavimai yra pateikiami *UML* panaudojimo atvejų diagramos pavidalu. 22 pav. yra pateikiama sistemos panaudojimo atvejų diagrama.



22 pav. *UML* panaudojimo atvejų diagrama

Panaudojimo atvejų diagramoje matoma, jog sistema yra sudaryta iš dviejų posistemių: autentifikacijos/autorizacijos bei paskyrų valdymo. Aktoriai atitinka sistemos naudotojus pagal jų roles.

Autentifikacijos ir autorizacijos posistemė yra atsakinga už prisijungimo funkcionalumą (angl. *flow*), kuris yra naudojamas prisijungiant tiek prie paskyrų valdymo sistemos, tiek prie *Kubernetes* valdymo skydelio. Prisijungimo funkcionalumas yra sudarytas iš kelių žingsnių: prisijungimo duomenų pateikimo bei konfigūruojamų patvirtinimų naudojant susietus mobiliojo ryšio įrenginius, kuriuos atlieka tiek prisijungiantysis naudotojas, tiek jo vadovas – vyr. inžinierius.

Paskyrų valdymo posistemė yra atsakinga už paskyrų sąrašo peržiūrą, naujų paskyrų kūrimą, egzistuojančių paskyrų nustatymų peržiūrą ir modifikavimą. Ši posistemė taip pat yra atsakinga už paskyros slaptažodžio keitimą, susieto mobiliojo ryšio įrenginio informacijos peržiūrą bei veiksmus, kurie gali būti atliekami su mobiliojo ryšio įrenginiu: susiejimą ir susiejimo atšaukimą.

2.2.2. Nefunkciniai reikalavimai

Kuriamam sprendimui yra keliami nefunkciniai reikalavimai:

1. Realizacijai
   1.1. *API* programų kodo padengimas vienetų testais turi siekti bent 85%;
   1.2. Autorizacijos procesas turi remtis *OAuth* 2.0 [58] autorizacijos protokolu;
   1.3. Autorizacijos procese turi būti naudojamas „Autorizacijos kodo“ (angl. *Authorization Code*) įgaliojimo (angl. *grant*) gavimo būdas ir *PKCE* išplėtimas [59];
   1.4. Mobiliojo ryšio įrenginio taikomoji programa turi veikti *Android* operacinėje sistemoje su 10 ir vėlesnėmis versijomis;
   1.5. Pranešimų siuntimui į mobiliojo ryšio įrenginio taikomąją programą turi būti naudojama *Firebase* platforma [60];
   1.6. Saityno programa turi teisingai veikti naudojant *Google Chrome* naršyklę su N-2 versijomis;
   1.7. Naudotojui turi būti suteikiama galimybė patvirtinti užklausą, atsiųstą į jo mobiliojo ryšio įrenginį, naudojant numatytąjį *Android* funkcionalumą, t.y. įvedant *PIN* kodą ar pateikiant piršto antspaudą;
   1.8. *API* prieigos taškai naudotojų užklausas turi apdoroti greičiau nei per 500 milisekundžių, kuomet sistema lygiagrečiai naudojasi 100 naudotojų;
2. Sistemos priežiūrai
   2.1. Programų kodas turi būti saugomas *Git* versijų valdymo sistemoje;
   2.2. Sistemos diegimas į veikimo aplinką turi būti kuo labiau automatizuotas;
   2.3. Sistemą galima diegti tiek į *Windows*, tiek į *UNIX* operacines sistemas naudojančius serverius;
   2.4. Visos sistemoje įvykusios klaidos, išimtys bei nesėkmingi autentifikacijos bandymai privalo būti užregistruoti sistemos žurnale;
3. Saugumui
   3.1. Sistema privalo būti atspari dažniausiai pasitaikantiems *XSS* ir *CSRF* atakų vektoriams, kurių apibūdinimai yra pateikiami *OWASP* šaltiniuose [61] [62];
   3.2. Kliento – serverio duomenų apsikeitimo kanalo apsauga turi būti pagrįsta pasirašytais *JWT* žetonais. Sistema turi naudoti dviejų tipų žetonus: prieigos (angl. *access*) bei atnaujinimo (angl. *refresh*). Atnaujinimo žetonas yra skirtas gauti naujam prieigos žetonui, kuomet senojo galiojimo trukmė pasibaigia. Atnaujinimo žetono trukmė apriboja sesijos gyvavimo trukmę. Pasibaigus galiojimo žetonui, naudotojas yra atjungiamas nuo sistemos. Naudotojui savarankiškai atsijungus nuo sistemos, atnaujinimo žetonas privalo būti pripažintas nebegaliojančiu;
   3.3. Prieigos žetonas privalo būti saugomas tik *HTTP* sausainėlyje, o *CSRF* žetonas – naršyklės sesijos talpykloje ir *HTTP* sausainėlyje;
   3.4. Sistemos veikimo aplinkoje tarp visų komunikuojančių komponentų privalo būti naudojamas *HTTPS* protokolas;
   3.5. Paskyrų slaptažodžiai privalo būti sumaišyti (angl. *hashed*) naudojant *bcrypt* [63] maišos algoritmą;
   3.6. Naudotojas, su administratoriaus pateiktu numatytuoju slaptažodžiu, kuris buvo nustatytas kuriant paskyrą, gali prisijungti tik prie paskyrų valdymo sistemos;

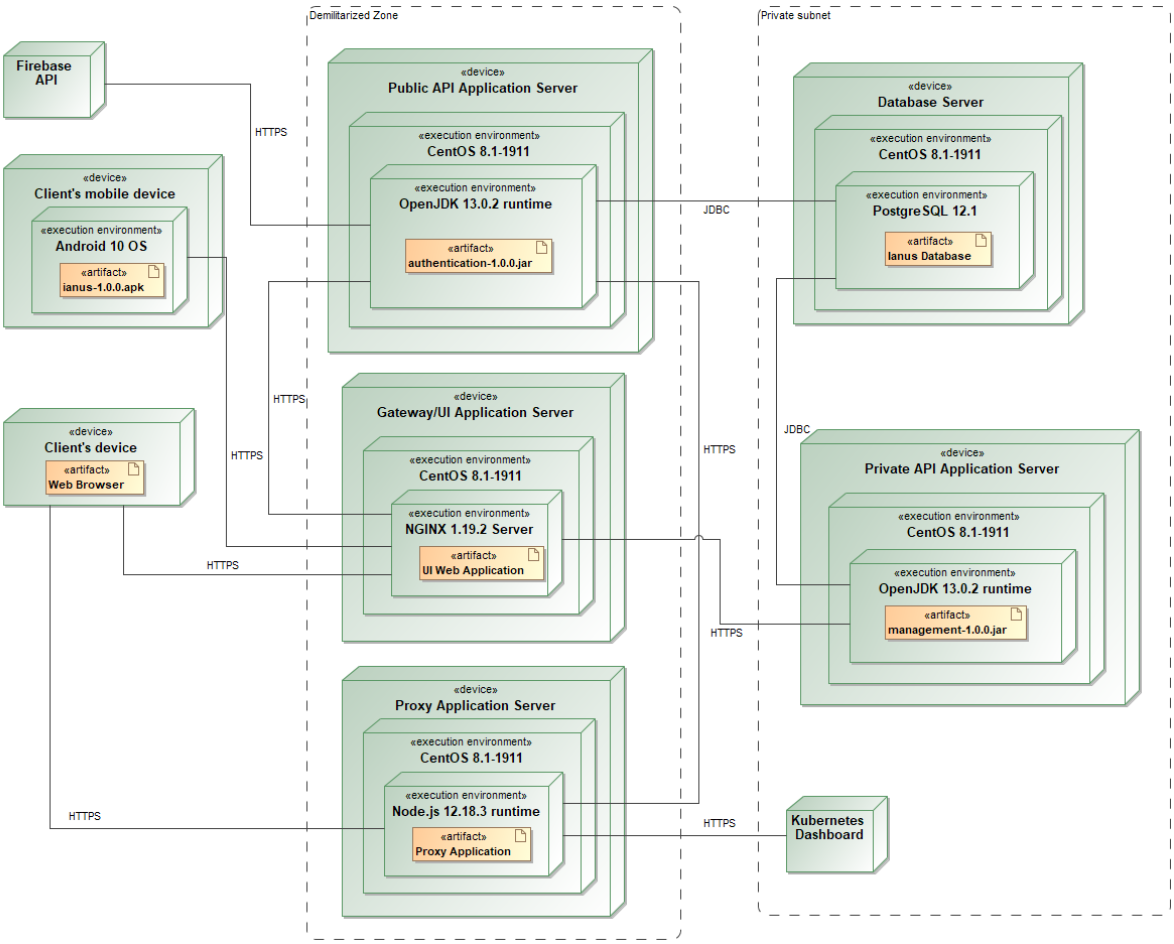
3.7. Naudotojui, pirmą kartą prisijungusiam prie paskyrų valdymo sistemos su administratoriaus nustatytu slaptažodžiu, turi būti pateikiamas pranešimas, kuriame būtų prašoma kuo greičiau pasikeisti numatytąjį slaptažodį;
3.8. Nesėkmingai atlikus 3 bandymus prisijungti prie paskyros, paskyra yra užblokuojama. Užblokuotą paskyrą atblokuoti gali tik administratorius;
3.9. Duomenų bazės duomenys privalo būti šifruojami (angl. *encryption of data at rest*);
3.10. Administratoriui sukūrus naują paskyrą, yra sukuriami 5 atsarginiai vienkartiniai slaptažodžiai, su kuriais naudotojas nelaimės atveju (pamiršus paskyros slaptažodį ar pametus susietą mobiliojo ryšio įrenginį bei jam sugedus) gali prisijungti prie paskyrų valdymo sistemos, apeinant pilną autentifikacijos procesą;
3.11. Autentifikacijos/autorizacijos užklausos, siunčiamos į susietą naudotojo mobiliojo ryšio įrenginį, privalo būti užšifruojamos naudojant naudotojo viešąjį raktą. Autentifikacijos/autorizacijos užklausų atsakymai serveriui turi būti pateikiami *JWT* formatu ir turi būti pasirašomi naudotojo privačiuoju raktu;
3.12. Paskyros kūrimo ar slaptažodžio keitimo metu, įvedamas slaptažodis turi būti bent 8 simbolių ilgio ir tenkinti aukščiausią saugumo lygį, taikant *zxcvbn* algoritmą;
3.13. Susiejant naudotojo mobiliojo ryšio įrenginį, jame turi būti sukuriamas 4096 bitų ilgio *RSA* raktas, kuris bus skirtas autentifikacijos/autorizacijos užklausų dešifravimui bei pasirašymui;
3.14. Vieningo prisijungimo sistemos ir naudotojo mobiliojo ryšio įrenginio kuriami *JWT* žetonai turi būti pasirašomi naudojant *RS512* algoritmą;

4. Kultūriniai
   4.1. Sistemos saityno programos ir mobiliojo ryšio įrenginio taikomosios programos naudotojo sąsajos turi būti pateikiamos anglų kalba;
5. Panaudojamumui
   5.1. Įvedamų duomenų validacijos klaidų pranešimai turi būti pateikiami iškart po įvedimo, raudonu šriftu, įvedimo lauko apačioje;
   5.2. Navigacijos meniu yra vizualiai išskiriamas aktyvus puslapis, kuriame tuo metu yra naudotojas;
   5.3. Kiekvieno saityno programos lango turinio sekcijos viršutinėje dalyje yra nurodomas 24 pikselių dydžio paryškintas lango pavadinimas;
   5.4. Formose naudojami duomenų pateikimo mygtukai yra vienodo dizaino, dydžio ir turi tokį patį išdėstymą;
   5.5. Naudotojui yra pateikiamas užimtumo indikatorius, kuomet sistema atlieka veiksmus, trunkančius ilgiau nei vieną sekundę;
   5.6. Naudotojui atliekant svarbius veiksmus, paspaudus veiksmo atlikimo mygtuką, turi būti pateikiamas patvirtinimo modalinis langas;
   5.7. Naudotojui atlikus veiksmus sistemoje, yra pateikiamas pranešimas apie sėkmingą arba nesėkmingą veiksmo atlikimą su paaiškinančiu pranešimu;
   5.8. Sistemos vartotojo sąsajoje turi būti naudojama *Ant* dizaino sistema [64].

2.3. Statinis sistemos modelis

2.3.1. Sistemos architektūra

23 pav. yra pateikiama *UML* diegimo diagrama, vaizduojanti fizinį sistemos išdėstymą testavimo ir produkcijos aplinkose.



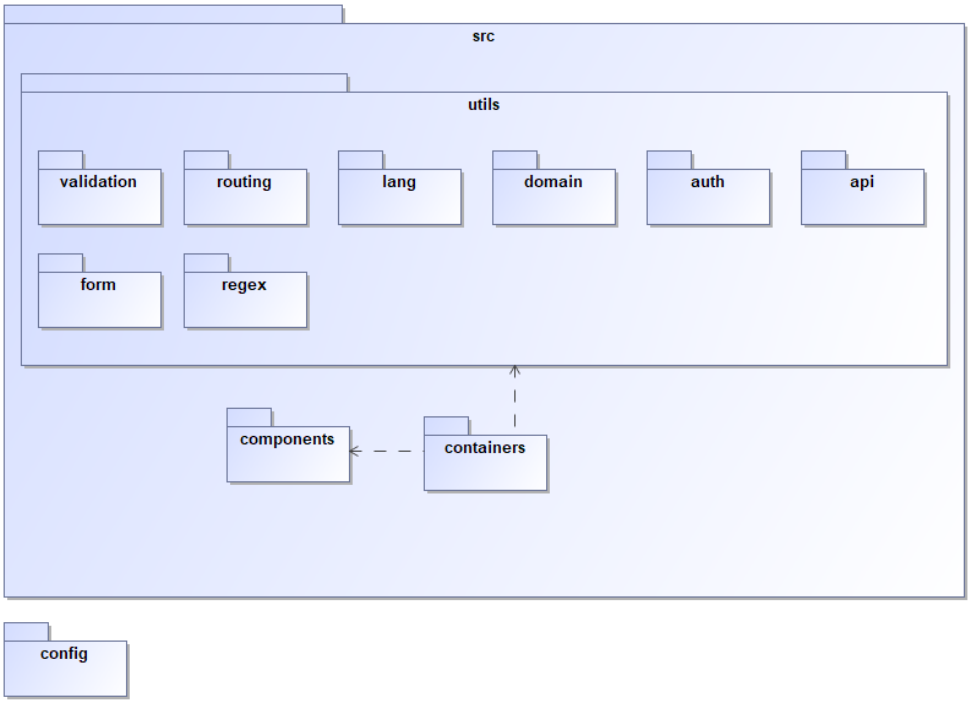
23 pav. *UML* diegimo diagrama

Sistemos komponentai yra diegiami į 5 serverius, kuriuose veikia *CentOS 8* operacinės sistemos:

- *Gateway/UI Application Server*: šiame serveryje yra diegiama saityno programa, kurią klientams teikia *NGINX* statinių failų serveris. Šis serveris taip pat perduoda užklausas į *API* programų prieigos taškus. Šis serveris yra talpinamas į demilitarizuotą tinklo zoną;
- *Database Server*: šiame serveryje veikia sistemos duomenų bazė. Abi *API* programos naudoja vieną bendrą duomenų bazę ir schemą. Šis serveris yra talpinamas į privatų tinklo potinklį;
- *Proxy Application Server*: šiame serveryje yra diegiama tarpinio įgaliotojo serverio tipo programa, kuri paslepia ir apsaugo tiesioginius kreipinius į *Kubernetes* valdymo skydelį. Ši programa pritaiko autentifikacijos bei autorizacijos mechanizmus, kurie apsaugo prieigą prie valdymo skydelio. Šis serveris yra talpinamas į demilitarizuotą tinklo zoną;
- *Private API Application Server*: šiame serveryje yra diegiama *Java* programa, kuri realizuoja paskyrų valdymo posistemę. Ši programa savyje turi jos veikimui reikalingą *Tomcat HTTP* programų serverį ir kreipiasi į duomenų bazės serverį. Šis serveris yra talpinamas į privatų tinklo potinklį;
- *Public API Application Server*: šiame serveryje yra diegiama *Java* programa, kuri realizuoja autentifikacijos/autorizacijos posistemę. Ši programa savyje turi jos veikimui reikalingą *Tomcat HTTP* programų serverį ir kreipiasi į duomenų bazės serverį bei į nutolusį *Firebase API* serverį. Šis serveris yra talpinamas į demilitarizuotą tinklo zoną.

2.3.2. Komponentų struktūra

24 pav. yra pateikta saityno programos paketų diagrama su pagrindiniais ryšiais tarp paketų.

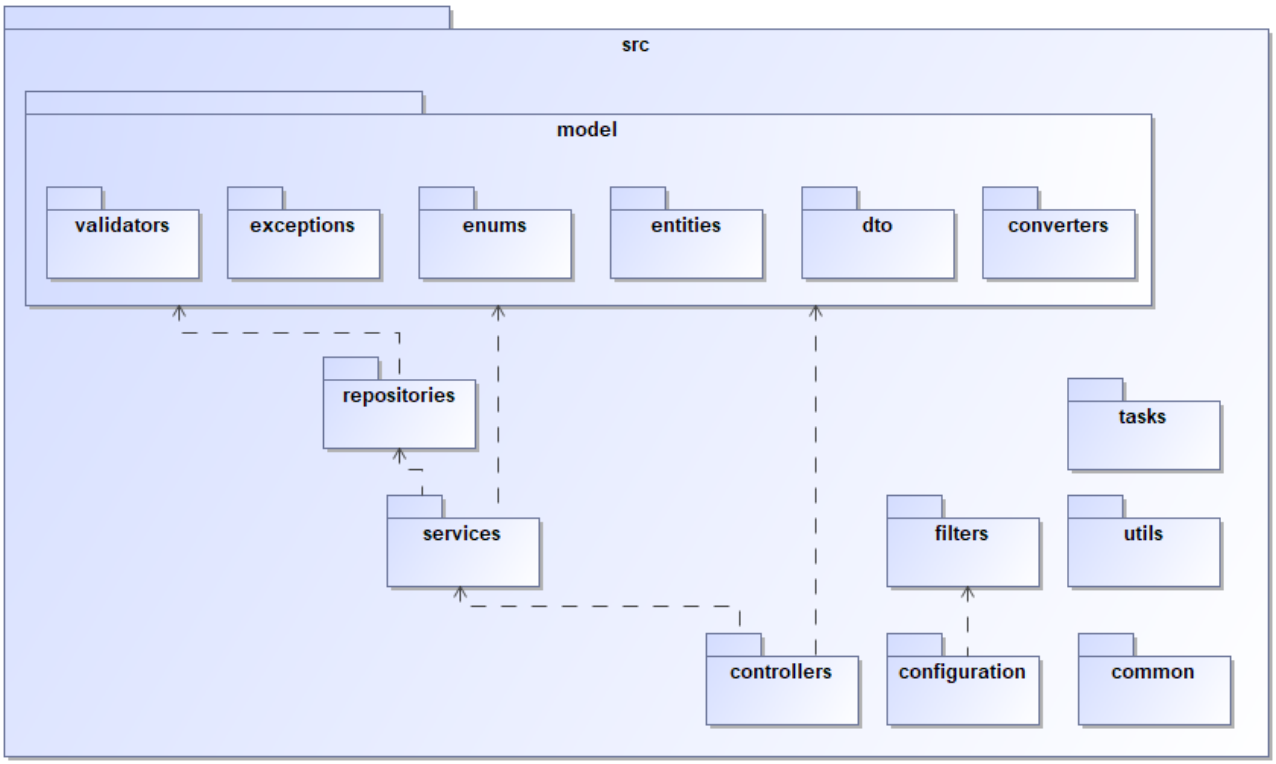


24 pav. *UML* paketų diagrama: saityno programos paketų struktūra

Saityno programos failai į paketus yra sugrupuoti pagal jų paskirtį:

- *utils* – Šiame pakete yra saugomos bendros funkcijos, skirtos komunikavimui su *API* serveriu, konfigūruojant programos kelius (angl. *routes*), vykdant formų validaciją bei valdant autentifikacijos/autorizacijos funkcionalumą;
- *config* – Šiame pakete yra talpinami failai, skirti *Webpack* pakavimo įrankio konfigūracijai;
- *components* – Šiame pakete yra saugomi pakartotinai naudojami *React* komponentai;
- *containers* – Šiame pakete yra saugomi konkrečių programos puslapių *React* komponentai.

Abiejų *API* programų paketų struktūra yra apibendrinta. 25 pav. yra pateikta *API* programos paketų diagrama su pagrindiniais ryšiais tarp paketų.

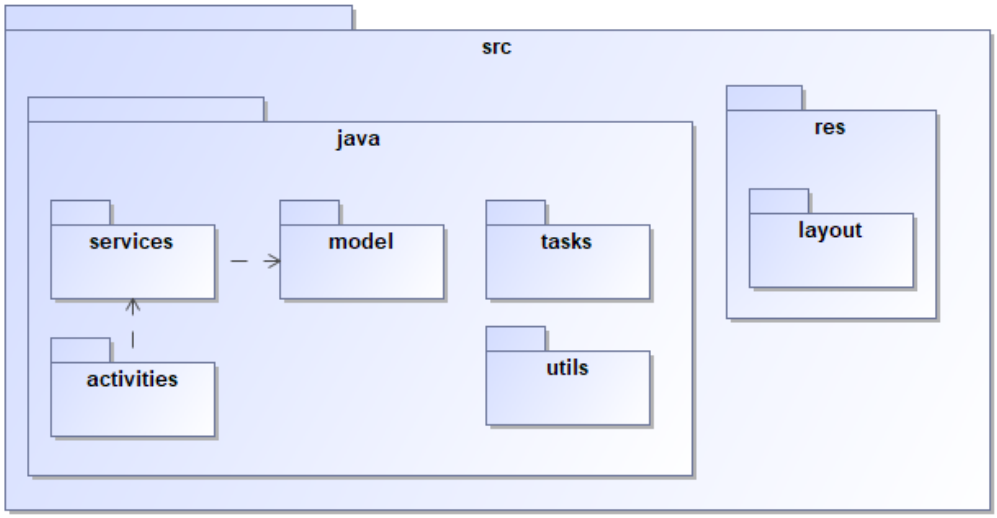


25 pav. *UML* paketų diagrama: apibendrinta *API* programos paketų struktūra

API programos klasės į paketus yra sugrupuotos pagal jų paskirtį:

- *entities* – Šiame pakete yra saugomos klasės, atitinkančios duomenų bazės esybes;
- *validators* – Šiame pakete yra saugomos klasės, skirtos objektų duomenų tikrinimui;
- *enums* – Šiame pakete yra saugomos išvardijamo tipo klasės;
- *dto* – Šiame pakete yra saugomos duomenų perdavimo klasės;
- *exceptions* – Šiame pakete yra saugomos išimčių klasės;
- *common* – Šiame pakete yra saugomos bendrai naudojamos klasės;
- *converters* – Šiame pakete yra saugomos klasės, skirtos konvertuoti vieno tipo objektus į kito tipo objektus;
- *controllers* – Šiame pakete yra saugomos valdiklių klasės, realizuojančios prieigos taškus;
- *services* – Šiame pakete yra saugomos klasės, skirtos dalykinės srities logikos realizacijai;
- *repositories* – Šiame pakete yra saugomos sąsajos, skirtos esybių perdavimui į/iš duomenų bazės;
- *tasks* – Šiame pakete yra saugomos klasės, kurios vykdo periodiškai suplanuotas užduotis;
- *utils* – Šiame pakete yra saugomos pagalbinės klasės;
- *filters* – Šiame pakete yra saugomos klasės, skirtos *HTTP* užklausų filtravimui;
- *configuration* – Šiame pakete yra saugomos programos konfigūracijos klasės.

26 pav. yra pateikta mobiliojo ryšio įrenginio taikomosios programos paketų diagrama su pagrindiniais ryšiais tarp paketų.

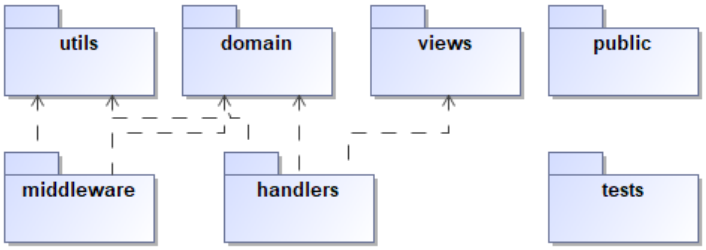


26 pav. *UML* paketų diagrama: mobiliojo ryšio įrenginio taikomosios programos paketų struktūra

Programos failai į paketus yra sugrupuoti pagal jų paskirtį:

- *layout* – Šiame pakete yra saugomi programos langų *XML* konfigūraciniai failai;
- *model* – Šiame pakete yra saugomos duomenų modelių klasės;
- *services* - Šiame pakete yra saugomos klasės, skirtos dalykinės srities logikos realizacijai;
- *activities* – Šiame pakete yra saugomos klasės, susiejančios *UI* elementus, dalykinės srities funkcionalumą bei programos lango funkcionalumą;
- *tasks* – Šiame pakete yra saugomos asinchroninių užduočių klasės;
- *utils* – Šiame pakete yra saugomos pagalbinės klasės.

27 pav. yra pateikta tarpinio įgaliotojo serverio programos paketų diagrama su pagrindiniais ryšiais tarp paketų.



27 pav. *UML* paketų diagrama: tarpinio įgaliotojo serverio programos paketų struktūra

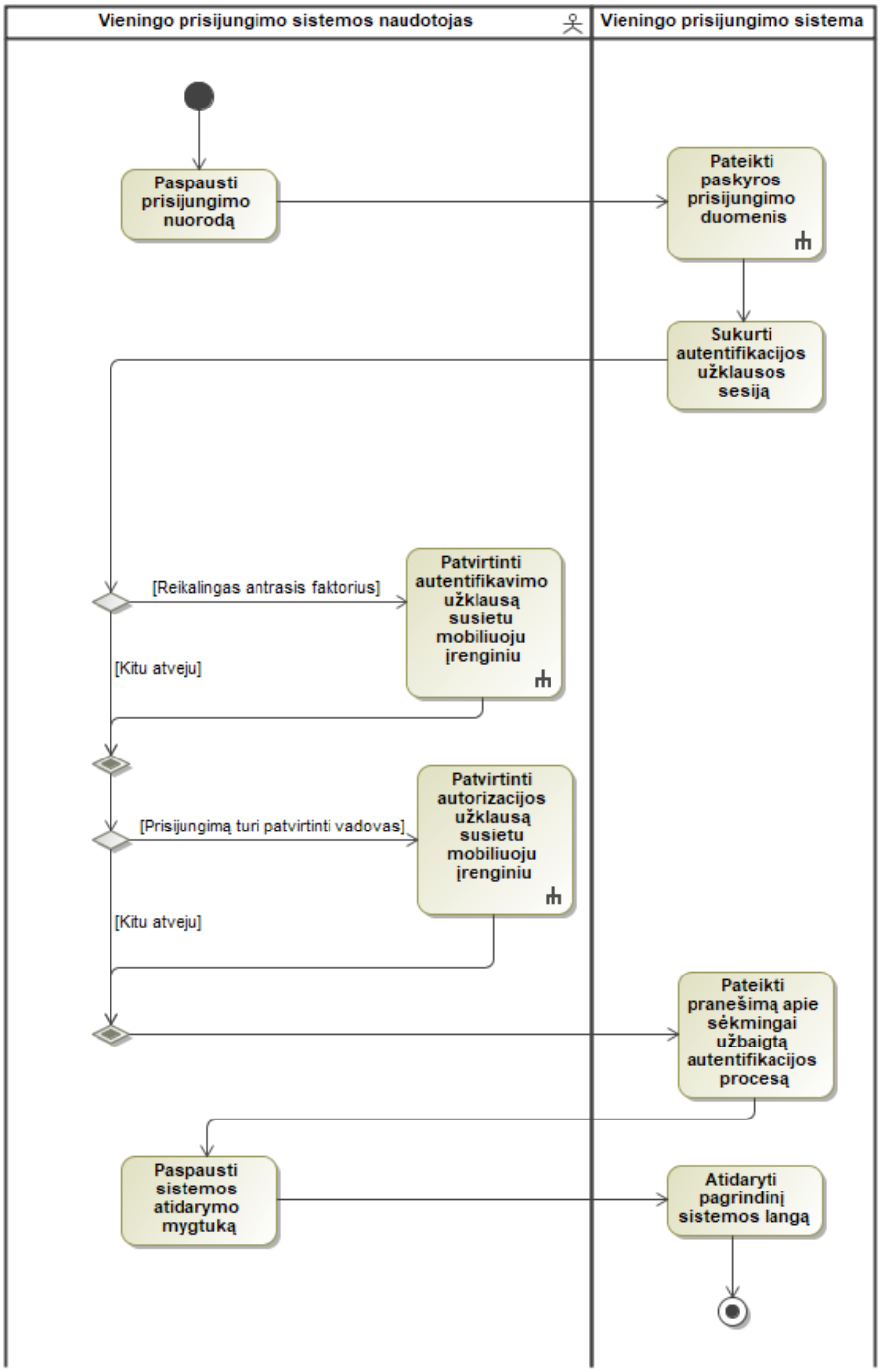
Programos failai į paketus yra sugrupuoti pagal jų paskirtį:

- *handlers* - Šiame pakete yra saugomos valdiklių funkcijos, realizuojančios prieigos taškus;
- *middleware* - Šiame pakete yra saugomos funkcijos, filtruojančios bei apdorojančios *HTTP* užklausas;
- *domain* – Šiame pakete yra saugomos funkcijos, skirtos dalykinės srities logikos realizacijai;
- *utils* – Šiame pakete yra saugomos pagalbinės funkcijos;
- *views* – Šiame pakete yra saugomi *HTML* puslapių šablonai;
- *tests* – Šiame pakete yra saugoma testams reikalinga programos konfigūracija;
- *public* – Šiame pakete yra saugomi statiniai resursų failai.

2.4. Dinaminis sistemos modelis

Dinaminis sistemos modelis yra pateikiamas *UML* veiklos diagramomis. Šios diagramos dokumentuoja sistemos panaudojimo atvejus. 28 pav. - 34 pav. yra pateiktos autentifikacijos/autorizacijos posistemės veiklos diagramos.

28 pav. pateikta prisijungimo prie sistemos veiklos diagrama.



28 pav. *UML* veiklos diagrama: Prisijungti prie sistemos

Sistemos naudotojas, norėdamas prisijungti prie sistemos (*Kubernetes* valdymo skydelio ar paskyrų valdymo sistemos), pirmiausiai turi pateikti paskyros prisijungimo duomenis (šio funkcionalumo veiklos diagrama pateikiama 29 pav.). Jei pateikti duomenys yra teisingi, sistema

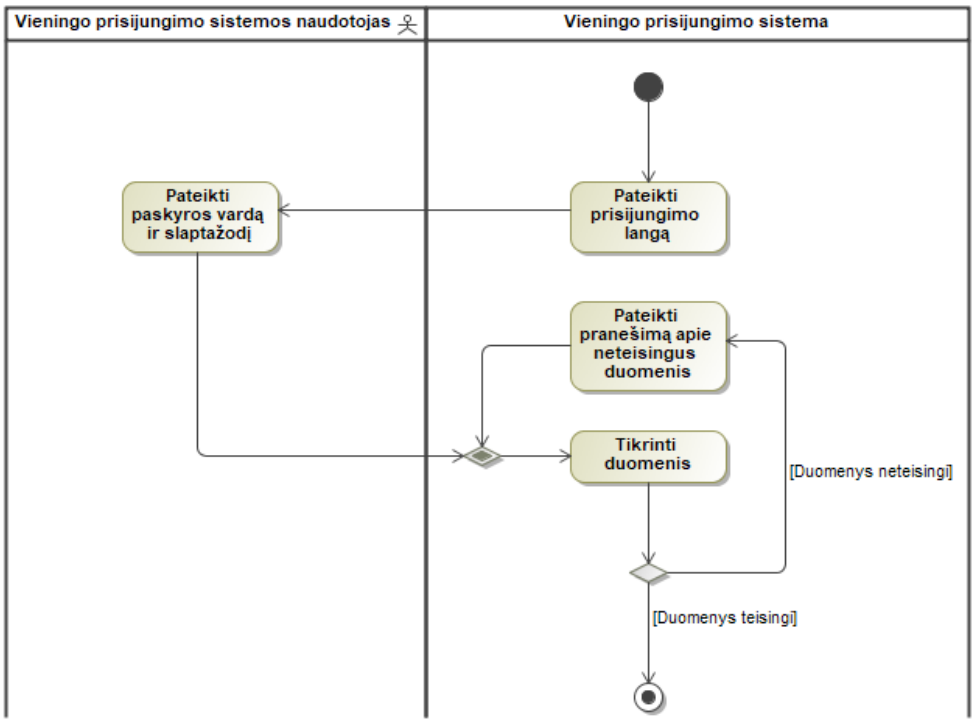
sukuria autentifikacijos užklausos sesiją, kuri yra apribota laike. Jei naudotojas nespėja sėkmingai užbaigti autentifikacijos proceso per sesijai skirtą laiką - naudotojas turi pradėti procesą iš naujo.

Tuomet pagal paskyros nustatymus yra tikrinama, ar sistema, prie kurios jungiasi naudotojas, reikalauja antrojo faktoriaus panaudojimo. Jei taip - naudotojas vykdo autentifikavimo užklausos patvirtinimo procesą, kurio veiklos diagrama pateikta 30 pav.

Sėkmingai patvirtinus autentifikavimo užklausą, yra tikrinama, ar sistema, prie kurios jungiasi naudotojas, reikalauja trečiojo faktoriaus panaudojimo. Jei taip - naudotojo vadovas vykdo autorizacijos užklausos patvirtinimo procesą, kurio veiklos diagrama pateikta 31 pav.

Sėkmingai patvirtinus autorizacijos užklausą, autentifikacijos procesas yra sėkmingai užbaigiamas ir naudotojui yra pateikiamas tai patvirtinantis pranešimas. Tuomet naudotojas gali paspausti sistemos atidarymo mygtuką ir pradėti naudotis sistema.

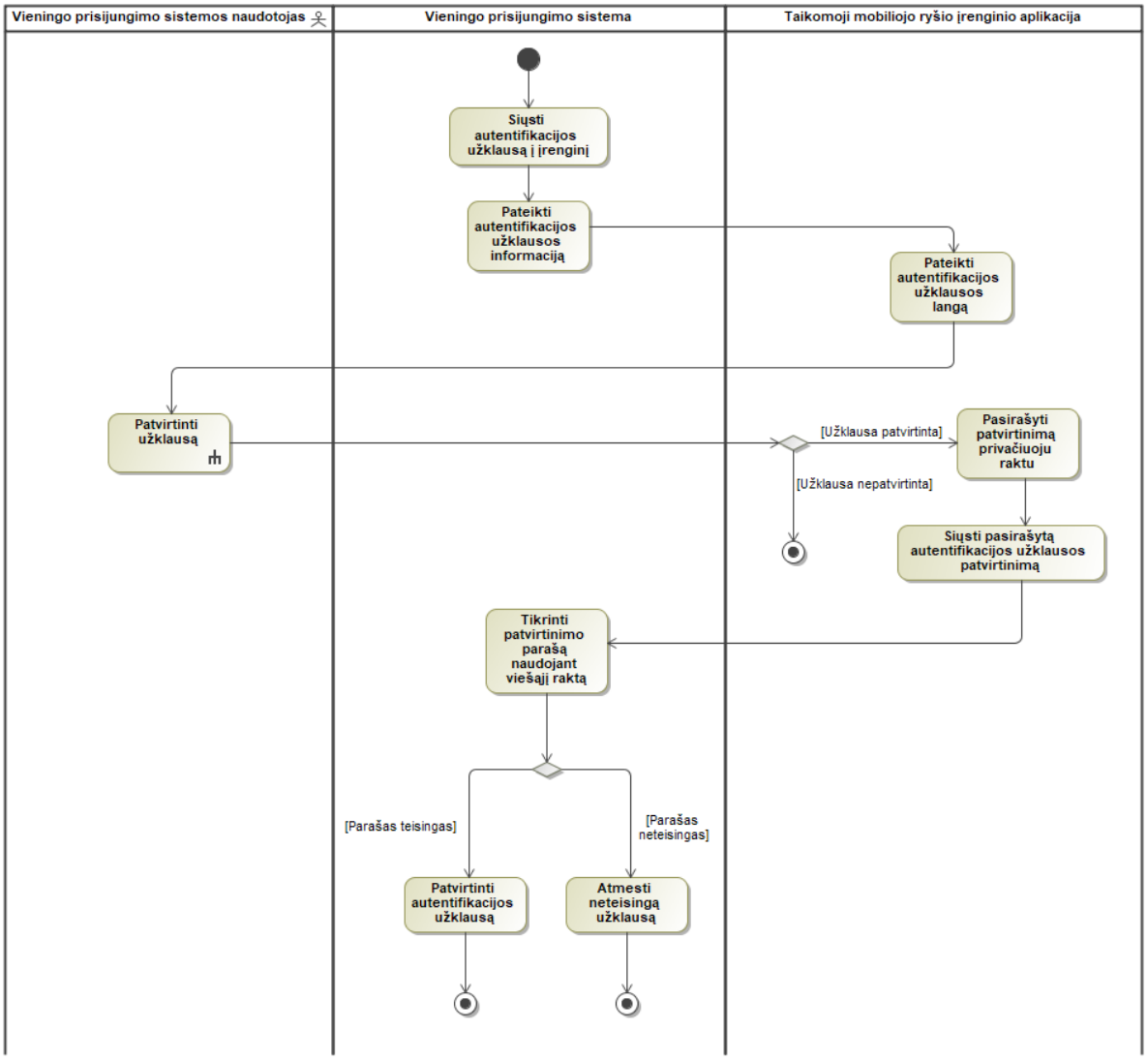
29 pav. pateikta paskyros prisijungimo duomenų pateikimo veiklos diagrama.



29 pav. *UML* veiklos diagrama: Pateikti paskyros prisijungimo duomenis

Naudotojui pradedant autentifikacijos procesą, sistema jam pateikia prisijungimo langą. Jame naudotojas pirmiausiai turi pateikti paskyros vardą ir slaptažodį. Jei naudotojas pateikia teisingus paskyros duomenis, autentifikacijos procesas yra tęsiamas toliau. Naudotojui pateikus neteisingus duomenis, sistema pateikia klaidos pranešimą.

30 pav. pateikta autentifikavimo užklausos patvirtinimo susietu mobiliuoju įrenginiu veiklos diagrama.

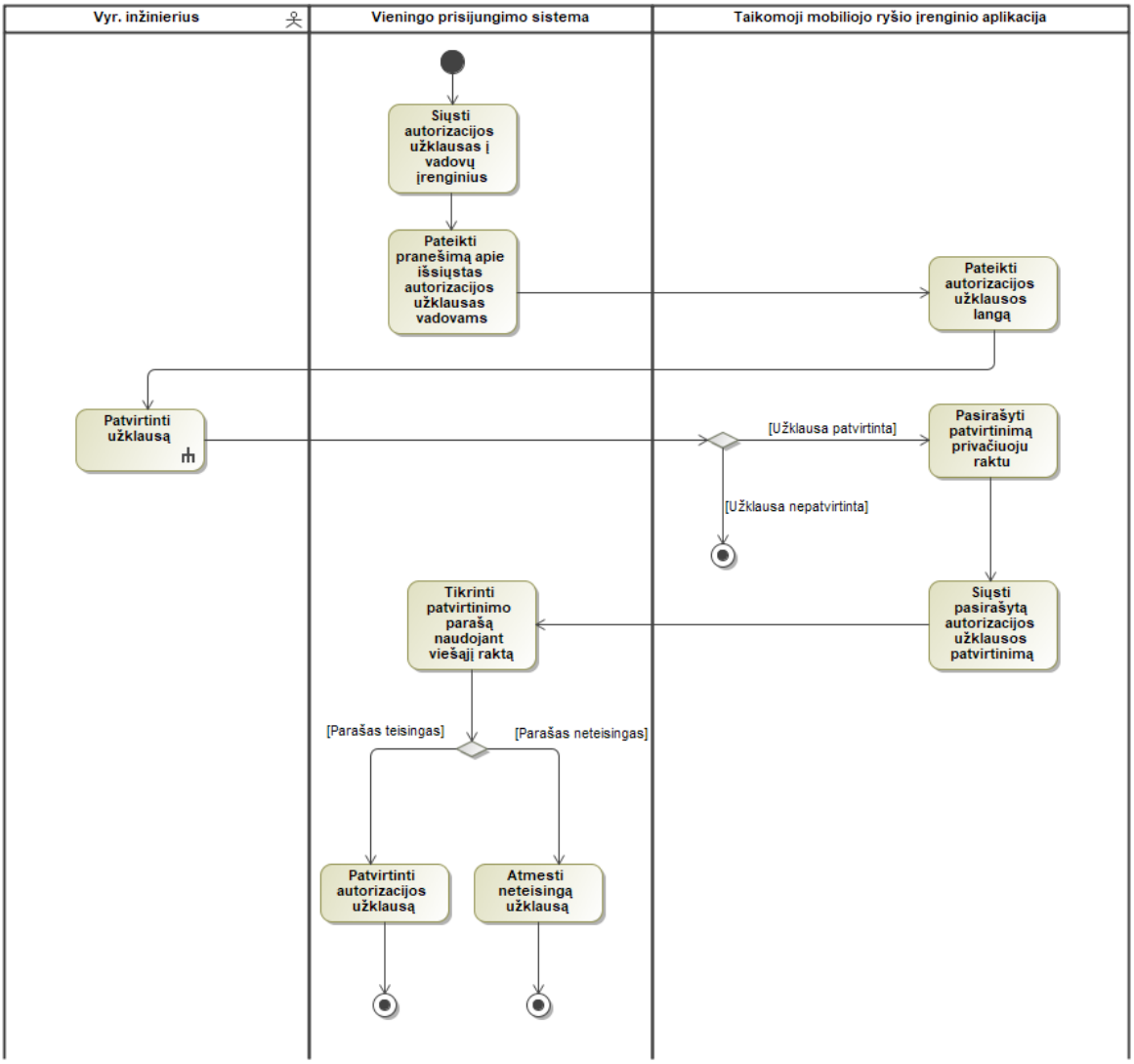


30 pav. *UML* veiklos diagrama: Patvirtinti autentifikavimo užklausą susietu mobiliuoju įrenginiu

Naudotojui vykdant autentifikacijos užklausos patvirtinimo procesą, sistema pirmiausiai išsiunčia užklausą į susietą naudotojo mobiliojo ryšio įrenginį ir saityno programoje pateikia autentifikacijos užklausos informaciją. Mobiliajame ryšio įrenginyje esanti taikomoji programa priima užklausą ir pateikia naudotojui autentifikacijos užklausos langą. Naudotojas tuomet vykdo užklausos patvirtinimo veiklą, kurios diagrama pateikiama 32 pav.

Jei patvirtinimo veiklos vykdymo metu užklausa nėra patvirtinama, tuomet autentifikacijos veikla yra užbaigiama nesėkmingai. Jei užklausa yra patvirtinama sėkmingai, tuomet mobiliajame įrenginyje yra suformuojamas autentifikacijos užklausos patvirtinimas, kuris yra pasirašomas naudotojo privačiuoju kriptografiniu raktu. Šis patvirtinimas tuomet yra siunčiamas vieningo prisijungimo sistemai. Sistema priima užklausos patvirtinimą ir jį tikrina panaudodama naudotojo viešąjį raktą. Jei patvirtinimas nėra teisingas – t.y. neatitinka naudotojo viešojo rakto, sistema atmeta neteisingą užklausą. Jei patvirtinimas yra teisingas, autentifikacijos užklausa yra patvirtinama ir autentifikacijos procesas yra tęsiamas toliau.

31 pav. pateikta autorizavimo užklausos patvirtinimo susietu mobiliuoju įrenginiu veiklos diagrama.

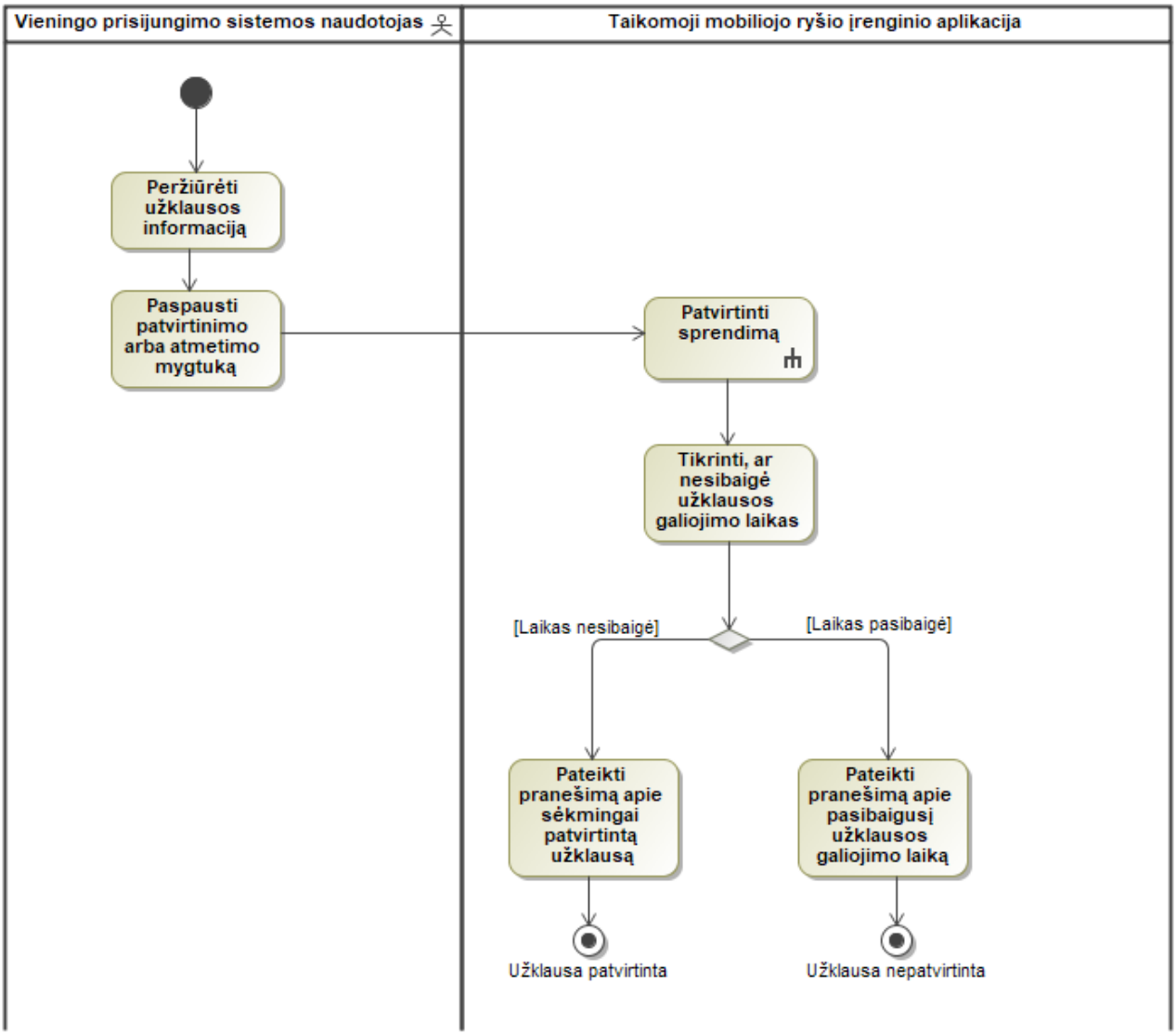


31 pav. *UML* veiklos diagrama: Patvirtinti autorizacijos užklausą susietu mobiliuoju įrenginiu

Vadovams vykdant autorizacijos užklausos patvirtinimo procesą, sistema pirmiausiai išsiunčia užklausas į susietus vadovų mobiliojo ryšio įrenginius, kurie priima užklausą ir pateikia vadovui autorizacijos užklausos langą. Vadovas tuomet vykdo užklausos patvirtinimo veiklą, kurios diagrama pateikiama 32 pav.

Jei patvirtinimo veiklos vykdymo metu užklausa nėra patvirtinama, tuomet autentifikacijos veikla yra užbaigiama nesėkmingai. Jei užklausa yra patvirtinama sėkmingai, tuomet mobiliajame įrenginyje yra suformuojamas autorizacijos užklausos patvirtinimas, kuris yra pasirašomas vadovo privačiuoju kriptografiniu raktu. Šis patvirtinimas tuomet yra siunčiamas vieningo prisijungimo sistemai. Sistema priima užklausos patvirtinimą ir jį tikrina panaudodama vadovo viešąjį raktą. Jei patvirtinimas nėra teisingas – t.y. neatitinka vadovo viešojo rakto, sistema atmeta neteisingą užklausą. Jei patvirtinimas yra teisingas, autentifikacijos užklausa yra patvirtinama ir autentifikacijos procesas yra tęsiamas toliau.

32 pav. pateikta užklausos patvirtinimo veiklos diagrama.

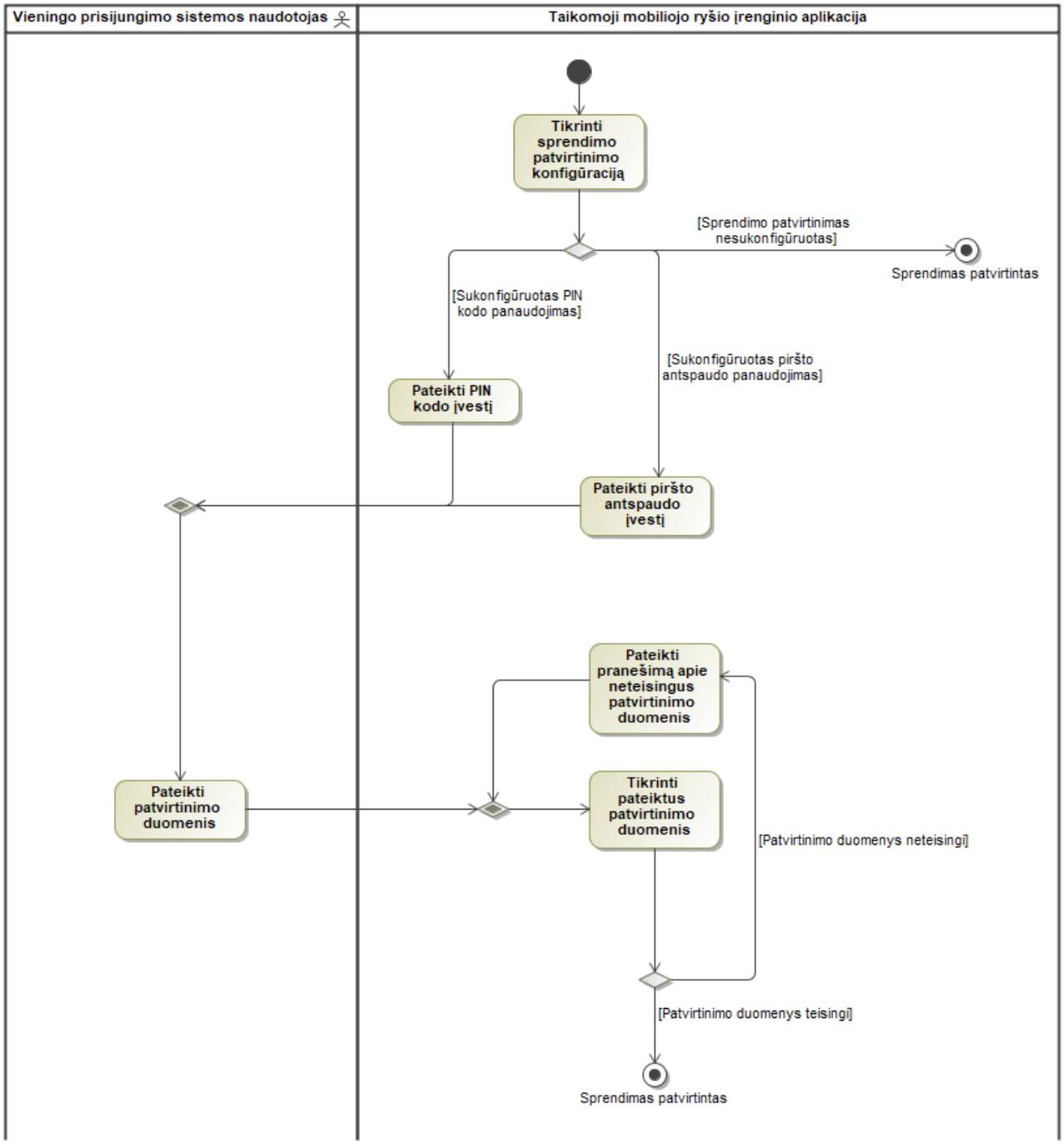


32 pav. *UML* veiklos diagrama: Patvirtinti užklausą

Pradėjus užklausos patvirtinimo veiklą, naudotojas peržiūri užklausos informaciją ir gali pasirinkti ar užklausa bus patvirtinta, ar atmesta. Pasirinkus veiksmą, naudotojas paspaudžia atitinkamą mygtuką ir vykdo sprendimo patvirtinimo veiklą, kurios diagrama yra pateikta 33 pav.

Patvirtinus sprendimą, yra tikrinama, ar nesibaigė užklausos galiojimo laikas. Jei laikas pasibaigė, yra pateikiamas atitinkamas klaidos pranešimas ir veikla yra užbaigiama nesėkmingai. Jei laikas nesibaigė, yra pateikiamas pranešimas apie sėkmingai patvirtintą užklausą ir veikla yra užbaigiama sėkmingai.

33 pav. pateikta sprendimo patvirtinimo veiklos diagrama.

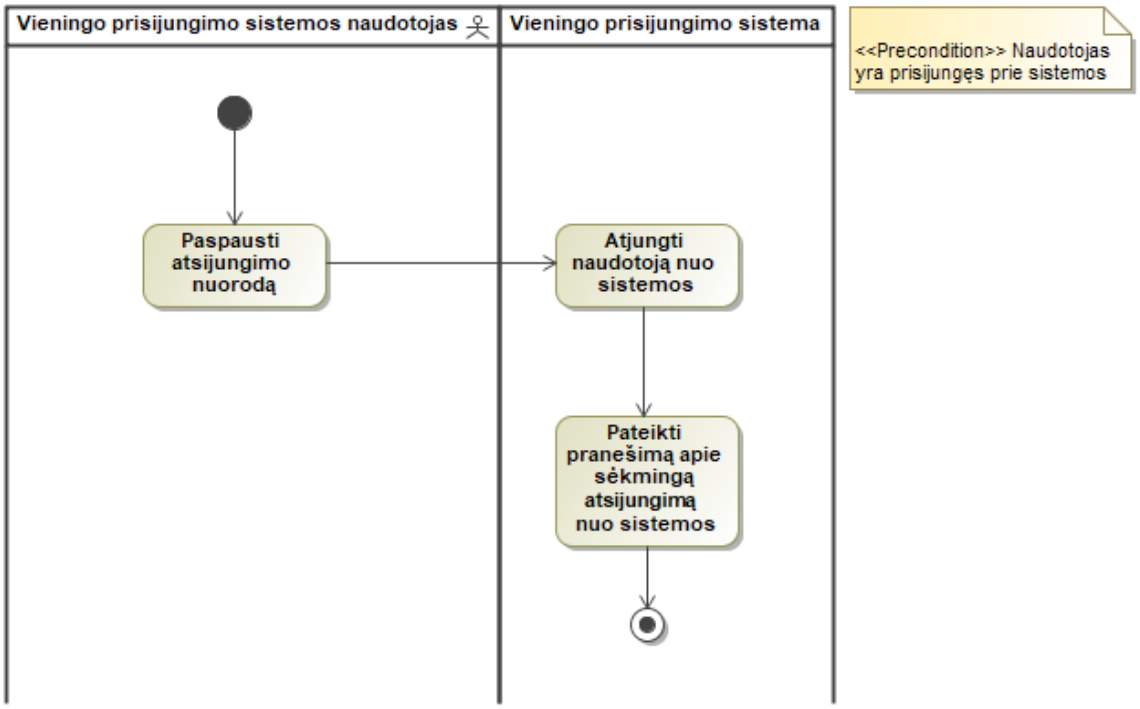


33 pav. *UML* veiklos diagrama: Patvirtinti sprendimą

Pradėjus sprendimo patvirtinimo veiklą, taikomoji mobiliojo ryšio įrenginio programa tikrina, ar yra sukonfigūruotas sprendimo patvirtinimo metodas. Jei metodas nėra sukonfigūruotas – sprendimas yra iškart patvirtinamas ir veikla yra užbaigiama.

Jei yra sukonfigūruotas PIN kodo ar pirštoantspaudo panaudojimas, įrenginys pateikia atitinkamą įvestį ir naudotojas pateikia patvirtinimo duomenis. Pateikus duomenis, įrenginys tikrina, ar pateiktas PIN kodas ar pirštoantspaudas yra teisingi. Jei duomenys nėra teisingi, įrenginys pateikia klaidos pranešimą. Jei duomenys yra teisingi, sprendimas yra patvirtinamas ir veikla yra užbaigiama.

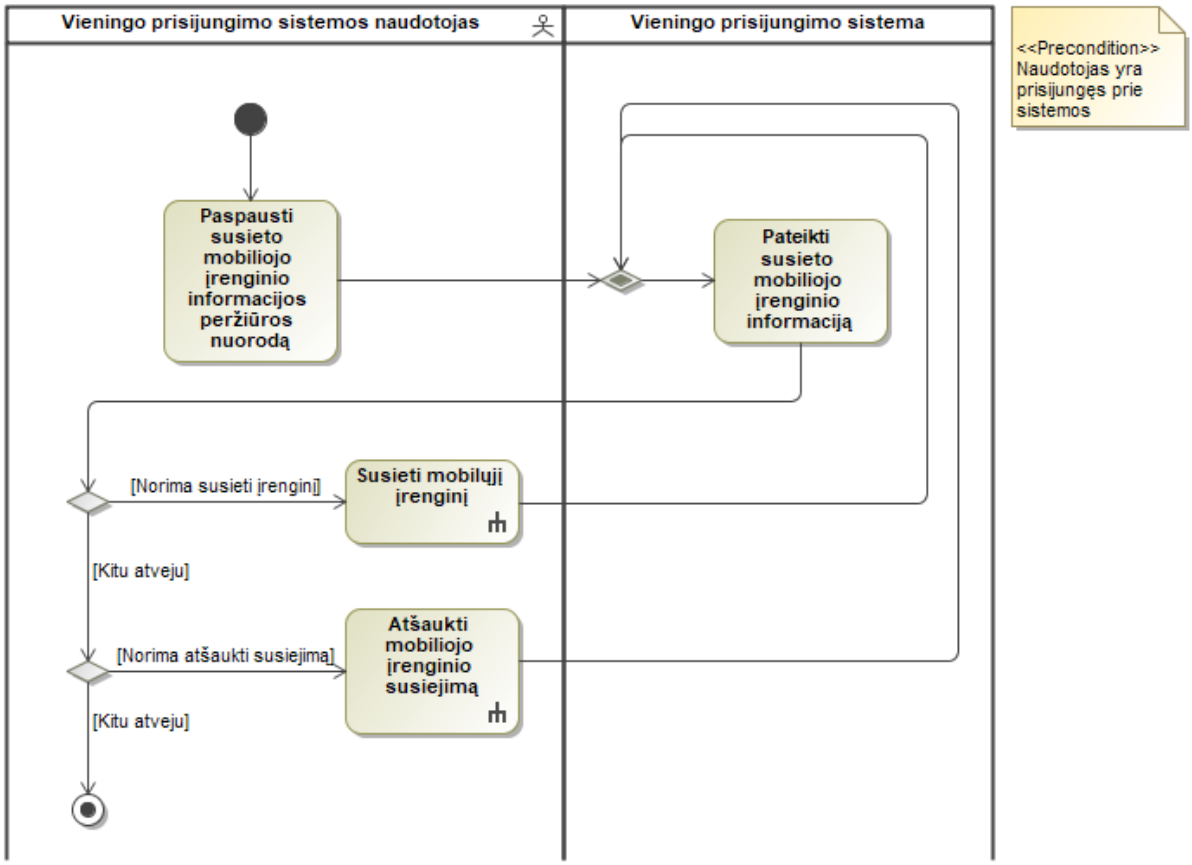
34 pav. pateikta atsijungimo nuo sistemos veiklos diagrama.



34 pav. *UML* veiklos diagrama: Atsijungti nuo sistemos

Prisijungęs naudotojas gali baigti darbą su sistema atsijungdamas nuo jos. Pabaigus darbą su sistema, naudotojui yra pateikiamas pranešimas apie sėkmingą atsijungimą nuo sistemos.

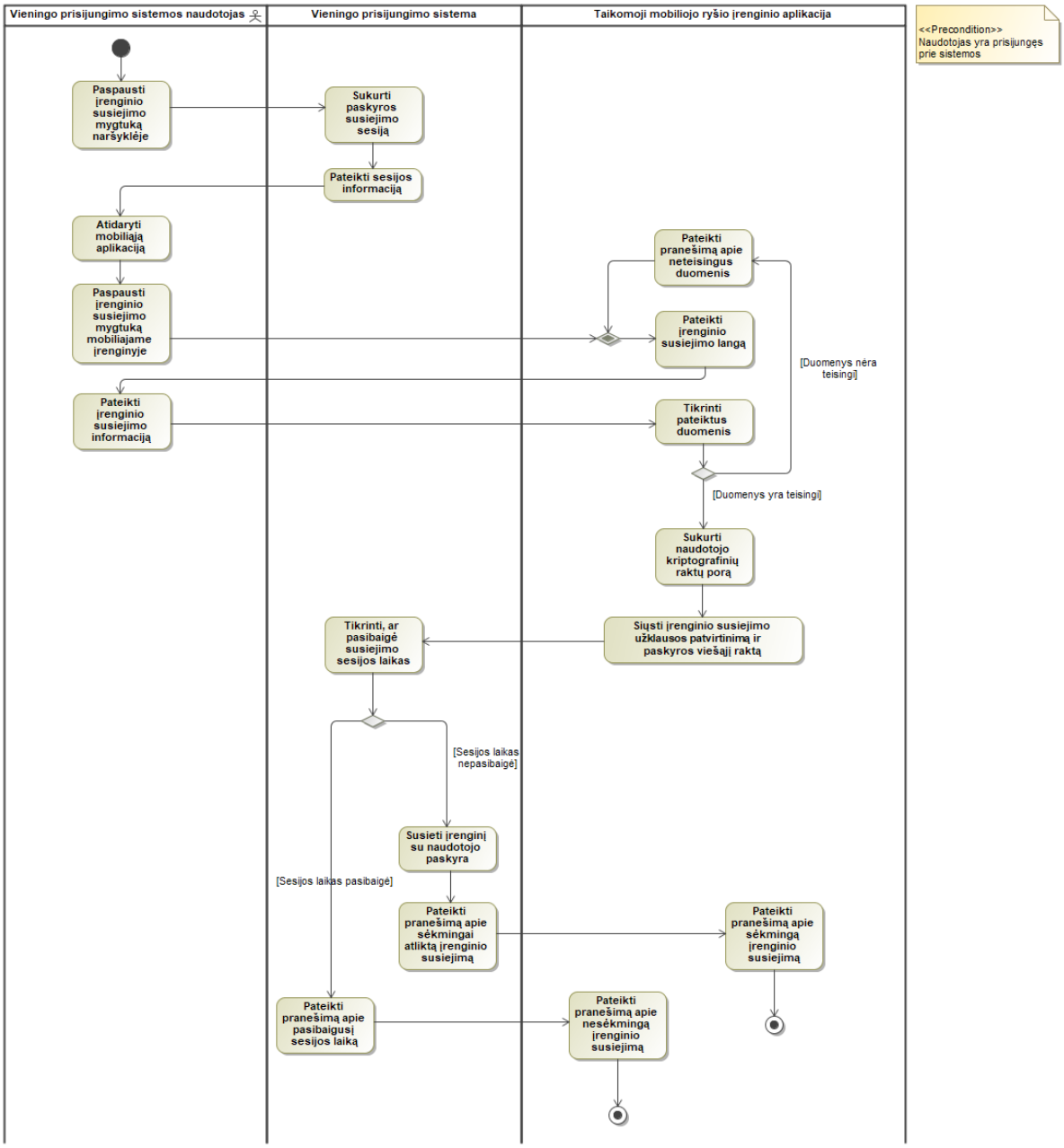
35 pav. - 42 pav. yra pateiktos paskyrų valdymo posistemės veiklos diagramos. 35 pav. pateikta susieto mobiliojo įrenginio informacijos peržiūros veiklos diagrama.



35 pav. *UML* veiklos diagrama: Peržiūrėti susieto mobiliojo įrenginio informaciją

Sistemos naudotojas gali peržiūrėti susieto mobiliojo įrenginio informaciją. Jei įrenginys nėra susietas, naudotojas gali pradėti įrenginio susiejimo procesą. Jei įrenginys yra susietas, naudotojas gali atšaukti susiejimą.

36 pav. pateikta mobiliojo įrenginio susiejimo veiklos diagrama.



36 pav. *UML* veiklos diagrama: Susieti mobilųjį įrenginį

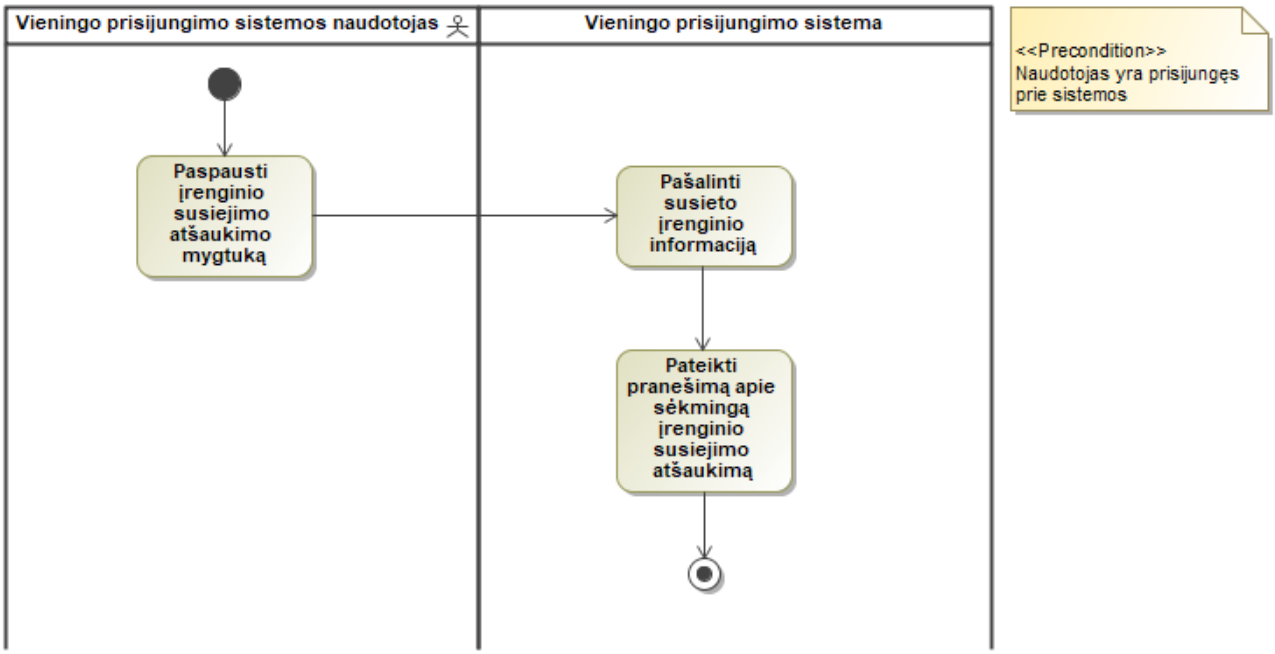
Naudotojas, norėdamas susieti mobiliojo ryšio įrenginį, pirmiausiai turi paspausti įrenginio susiejimo mygtuką saityno programoje. Tuomet naudotojui saityno programoje yra pateikiama sesijos informacija: likęs laikas ir žetonas, skirtas įrenginio susiejimui.

Tuomet naudotojas atidaro mobiliąją programą, paspaudžia įrenginio susiejimo mygtuką ir susiejimo formoje pateikia reikiamą informaciją: įrenginio vardą bei žetoną. Tuomet mobiliojo ryšio įrenginys tikrina pateiktus duomenis. Jei duomenys nėra teisingi, naudotojui mobiliojo ryšio įrenginyje yra pateikiamas klaidos pranešimas. Jei duomenys yra teisingi, naudotojo įrenginyje yra sukuriama kriptografinių raktų pora. Viešasis raktas kartu su įrenginio susiejimo užklausos patvirtinimu yra siunčiami vieningo prisijungimo sistemai. Sistema priima įrenginio susiejimo užklausą ir tikrina, ar sesijos galiojimo laikas nėra pasibaigęs. Jei naudotojas nespėja sėkmingai

užbaigti susiejimo proceso per sesijai skirtą laiką – pateikiamas pranešimas apie pasibaigusį sesijos laiką ir susiejimo veikla turi būti pradedama iš naujo.

Jei sesijos laikas nėra pasibaigęs, sistema sėkmingai susieja įrenginį su naudotojo paskyra. Susiejus įrenginį, veikla yra užbaigiama pateikiant pranešimus apie sėkmingą įrenginio susiejimą tiek saityno programoje, tiek taikomojoje mobiliojo ryšio įrenginio programoje.

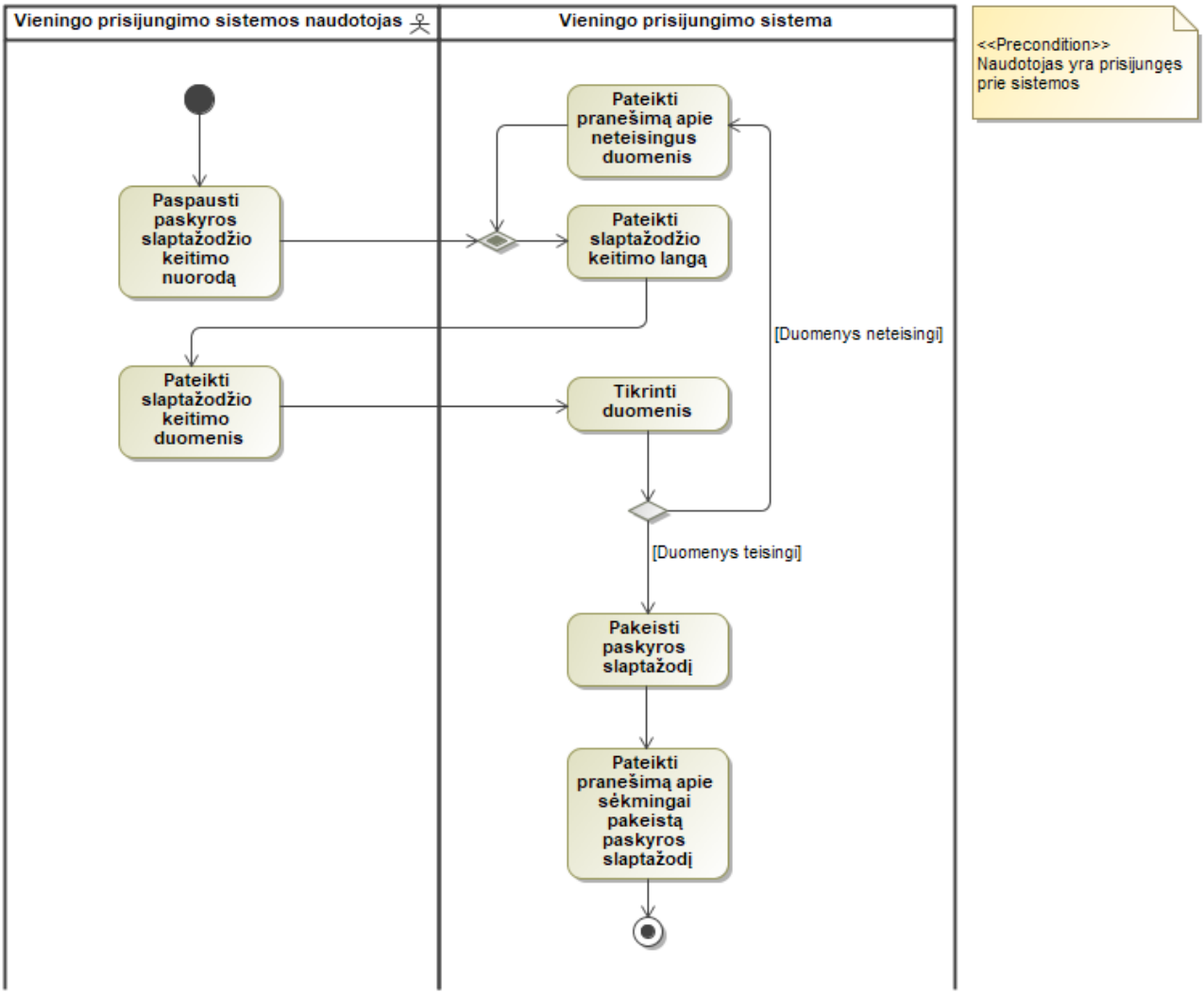
37 pav. pateikta mobiliojo įrenginio susiejimo atšaukimo veiklos diagrama.



37 pav. *UML* veiklos diagrama: Atšaukti mobiliojo įrenginio susiejimą

Naudotojas, norėdamas atšaukti mobiliojo ryšio įrenginio susiejimą, turi paspausti susiejimo atšaukimo mygtuką. Tuomet sistema pašalina susieto įrenginio informaciją ir galiausiai pateikia pranešimą apie sėkmingą įrenginio susiejimo atšaukimą.

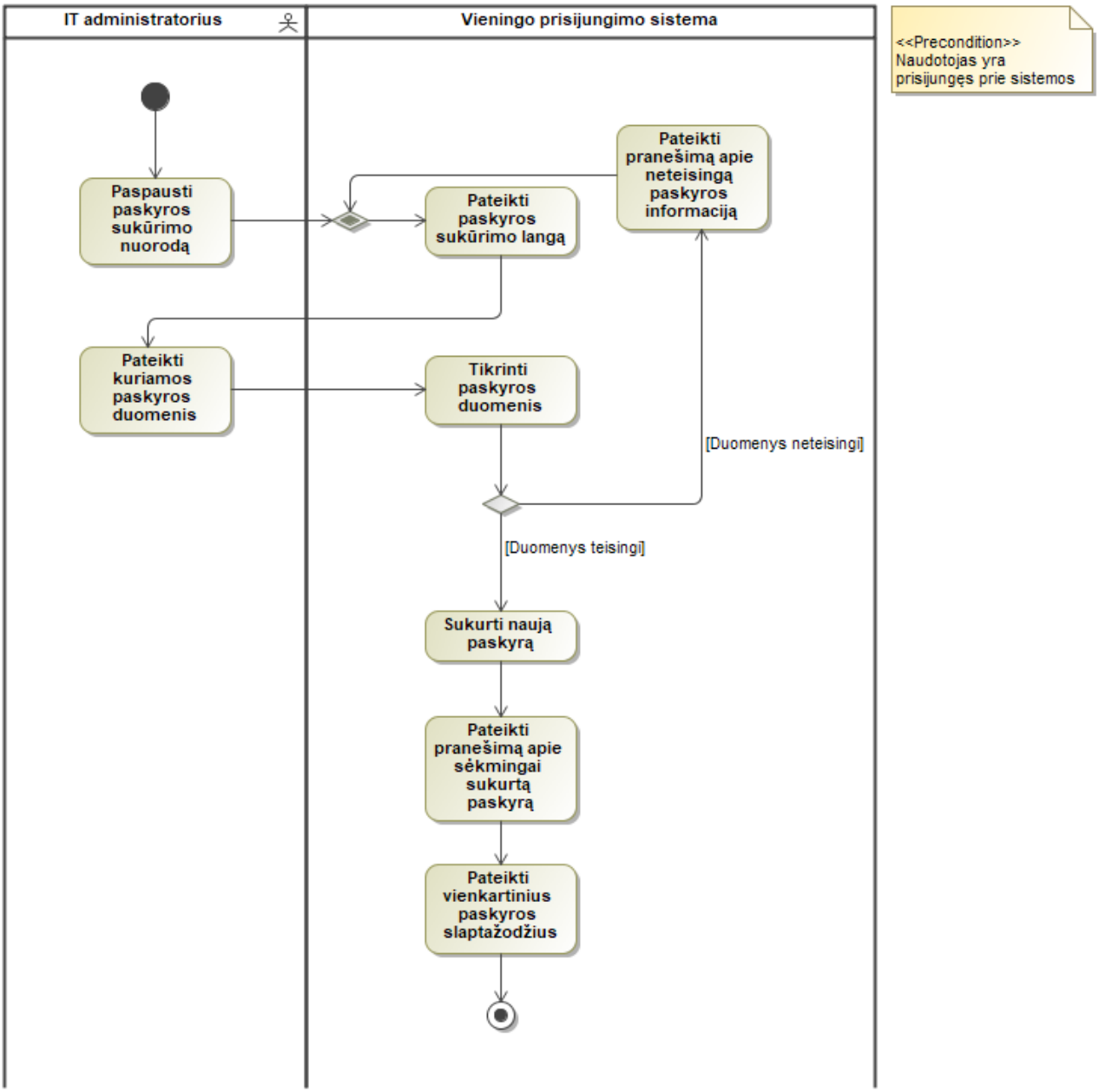
38 pav. pateikta paskyros slaptažodžio pakeitimo veiklos diagrama.



38 pav. *UML* veiklos diagrama: Pakeisti paskyros slaptažodį

Naudotojas gali pasikeisti savo paskyros slaptažodį užpildydamas slaptažodžio keitimo formą. Jei naudotojas pateikia neteisingą seną slaptažodį – jam yra pateikiamas pranešimas apie neteisingus duomenis. Jei senas slaptažodis yra teisingas – naudotojo paskyros slaptažodis yra pakeičiamas. Apie sėkmingą slaptažodžio pakeitimą naudotojas yra informuojamas sisteminiu pranešimu.

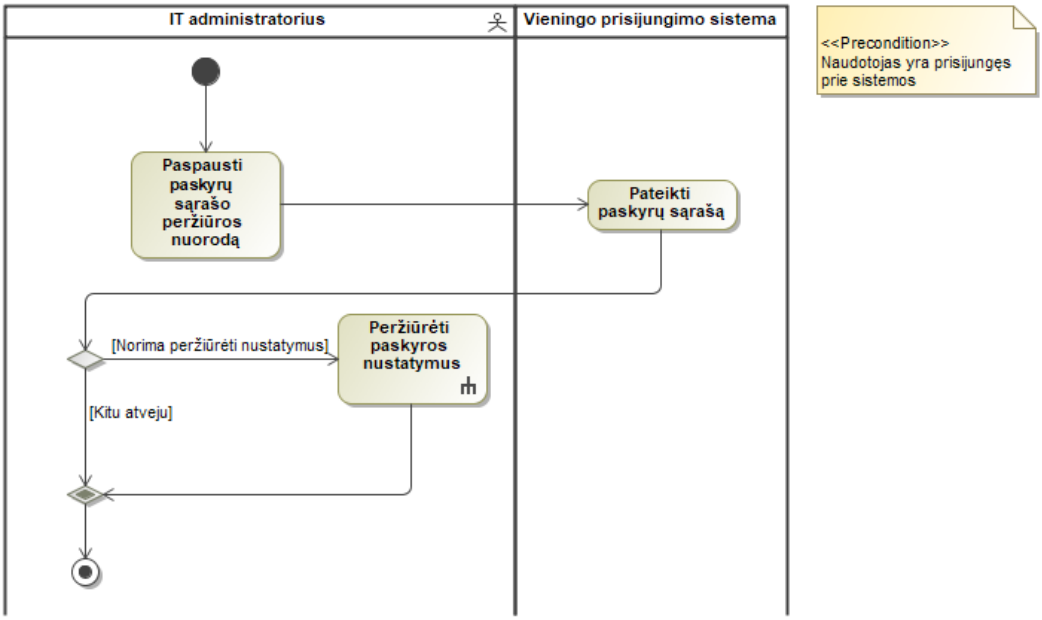
39 pav. pateikta naujos paskyros sukūrimo veiklos diagrama.



39 pav. *UML* veiklos diagrama: Sukurti naują paskyrą

Sistemos administratorius kitiems sistemos naudotojams gali sukurti naujas paskyras. Pateikus paskyros kūrimo formą yra tikrinami jos duomenys – paskyros vardo unikalumas, slaptažodžio ilgis ir stiprumas. Jei pateikti duomenys yra neteisingi, naudotojui yra pateikiamas pranešimas apie neteisingus duomenis. Jei pateikti paskyros duomenys yra teisingi – paskyra yra sukuriama. Apie sėkmingą paskyros sukūrimą administratorius yra informuojamas sisteminiu pranešimu. Taip pat sukūrus paskyrą administratoriui yra pateikiami 5 atsarginiai sugeneruoti vienkartiniai paskyros slaptažodžiai. Paskyros vardą, slaptažodį bei vienkartinius slaptažodžius administratorius paskyros savininkui turi perduoti rankiniu būdu – tiesiogiai sąveikaujant, su tikslu užtikrinti, jog informacija bus perduota saugiai ir nebus perimta pašalinių asmenų.

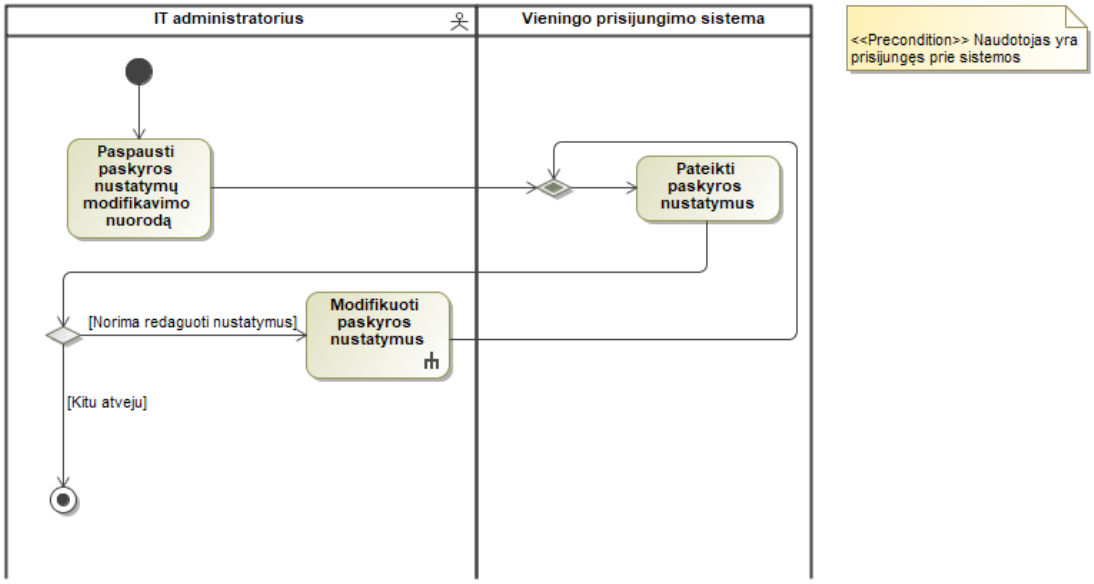
40 pav. pateikta paskyrų sąrašo peržiūros veiklos diagrama.



40 pav. *UML* veiklos diagrama: Peržiūrėti paskyrų sąrašą

Sistemos administratorius gali peržiūrėti visų paskyrų sąrašą. Paskyros gali būti atrenkamos pagal vardą. Sistemos administratorius taip pat gali peržiūrėti kiekvienos paskyros nustatymus (šio funkcionalumo veiklos diagrama pateikiama 41 pav.).

41 pav. pateikta paskyros nustatymų peržiūros veiklos diagrama.

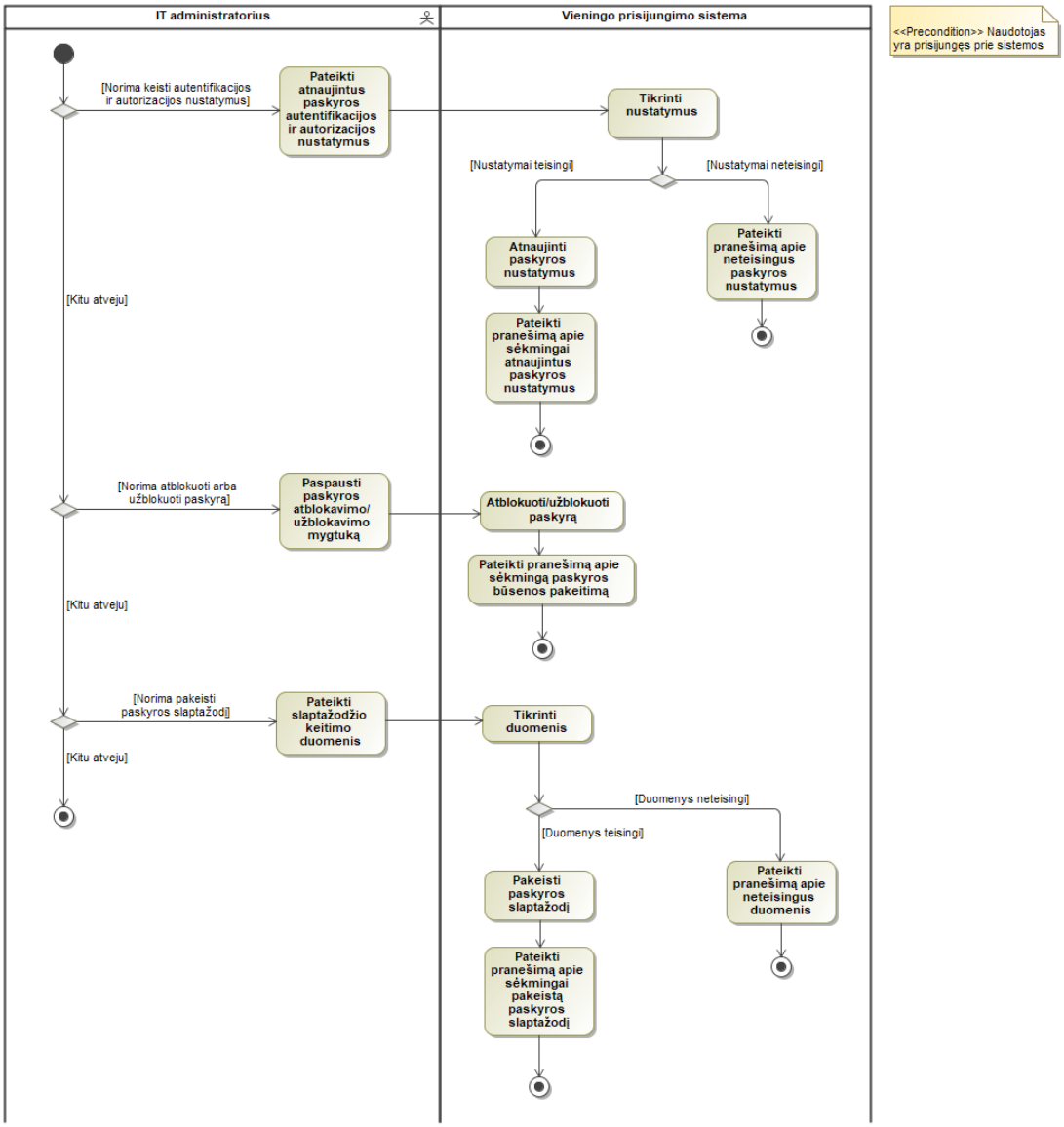


41 pav. *UML* veiklos diagrama: Peržiūrėti paskyros nustatymus

Sistemos administratorius gali peržiūrėti paskyros nustatymus, kuriuose yra nurodoma paskyros rolė bei autentifikacijos/autorizacijos konfigūracija skirtingoms sistemoms: antrojo bei trečiojo faktoriaus panaudojimas autentifikacijos proceso metu, jungiantis prie skirtingų sistemų. Sistemos

administratorius taip pat gali modifikuoti šiuos paskyros nustatymus (šio funkcionalumo veiklos diagrama pateikiama 42 pav.).

42 pav. pateikta paskyros nustatymų modifikavimo veiklos diagrama.



42 pav. *UML* veiklos diagrama: Modifikuoti paskyros nustatymus

Sistemos administratorius gali modifikuoti paskyros autentifikacijos/autorizacijos konfigūraciją skirtingoms sistemoms: prieigos suteikimą/uždraudimą, antrojo bei trečiojo faktoriaus panaudojimą autentifikacijos proceso metu. Jei pateikti nustatymai yra neteisingi, naudotojui yra pateikiamas pranešimas apie neteisingus nustatymus. Apie sėkmingą paskyros nustatymų atnaujinimą sistemos administratorius yra informuojamas sisteminiu pranešimu. Administratorius paskyros modifikavimo metu gali atblokuoti paskyrą, kuomet ši buvo užblokuota po 3 nesėkmingų bandymų prisijungti, arba užblokuoti aktyvią paskyrą. Administratoriui taip pat suteikiama galimybė pakeisti paskyros slaptažodį, kuomet naudotojas pamiršo pagrindinį slaptažodį ir pametė (ar išnaudojo) visus vienkartinius slaptažodžius.

2.5. Išvados

Projektinės dalies metu buvo suprojektuotas dviejų faktorių autentifikavimo ir autorizavimo metodas, kuris tenkintų kritinės infrastruktūros sistemoms ir nuotolinei prieigai prie jų keliamus saugos reikalavimus.

Vieningo prisijungimo bei tarpinio įgaliotojo serverio sistemų projektavimas, atsižvelgiant į keliamą nefunkcinį reikalavimą - autorizacijos procesą realizuoti naudojant *OAuth 2.0* protokolą - leidžia sukurti bendrai naudojamą vieningo prisijungimo ir identiteto valdymo sistemą, kurios viena iš klienčių būtų tarpinė įgaliotojo serverio sistema, teikianti nuotolinę prieigą prie *Kubernetes* valdymo skydelio.

Vieningo prisijungimo sistemos projektavimas, pagrįstas naudotojo autentifikacijos ir autorizacijos veiklos procesų modeliavimu, leido identifikuoti autentifikacijos proceso aspektus skirtinguose naudotojo identiteto valdymo etapuose: paties naudotojo tapatybės patvirtinime, autorizacijoje per nuotolinę prieigą. Priklausomai nuo autorizacijos užklausos patvirtinimo rezultato, naudotojui yra suteikiama arba nesuteikiama nuotolinė prieiga prie *Kubernetes* valdymo skydelio.

Suprojektuotas autentifikacijos veiklos modelis tenkina: 1) kritinės infrastruktūros sistemoms keliamą kelių kanalų (angl. *out-of-band authentication*) panaudojimo reikalavimą, kadangi yra naudojama tiek naudotojo kompiuteryje veikianti naršyklė, tiek naudotojo mobiliojo ryšio įrenginys; 2) autentifikacijos užklausos patvirtinimo (autorizacijos) reikalavimą, kadangi naudotojui nuotolinė prieiga yra suteikiama tik tuo atveju, jei nuotolinės prieigos prašymo užklausą patvirtina jo vadovas; 3) trečiąjį reikalavimą – sistemos komponentai tiek tarpusavyje, tiek su naudotojo įrenginiais komunikuoja naudodami patikimą informacijos perdavimo kanalą: *HTTPS* protokolą.

Sudaryta sprendimo koncepcija, funkciniai ir nefunkciniai reikalavimai bei veiklos ir diegimo modeliai sąlygoja tolimesnį sistemos realizacijos etapą.

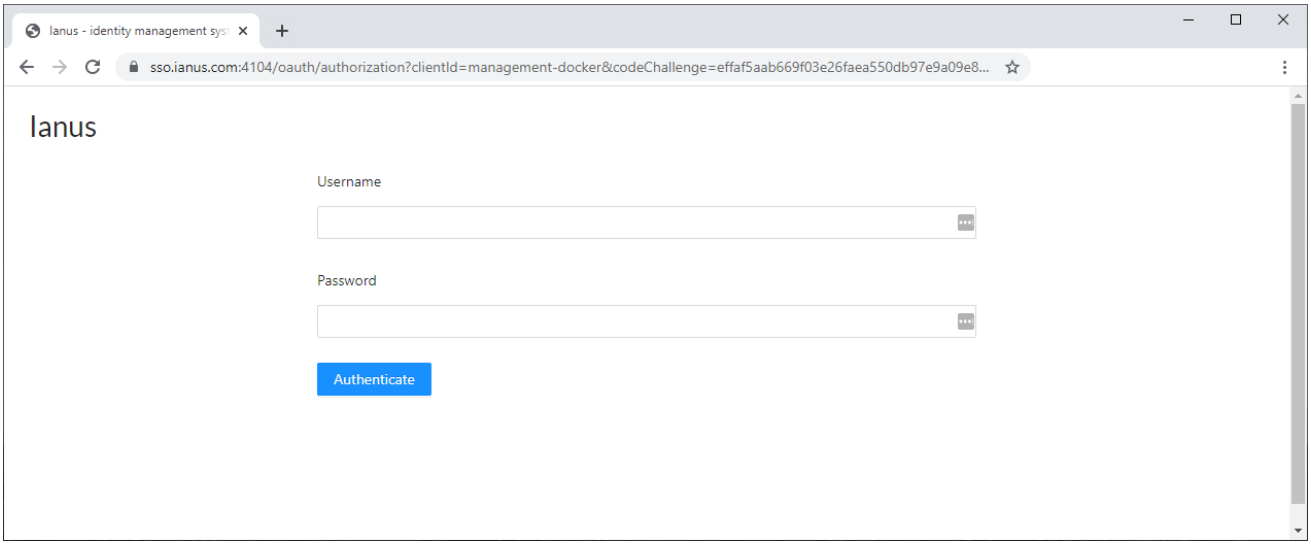
3. Kelių faktorių autentifikacijos metodo ir jį palaikančios sistemos realizacija

Realizacijos dalies tikslas – remiantis projektu, realizuoti sistemą, kuri valdytų naudotojų paskyras, realizuotų kelių faktorių autentifikacijos ir autorizacijos metodą bei suteiktų nuotolinę prieigą prie *Kubernetes* valdymo skydelio. Šiam tikslui pasiekti yra keliami šie uždaviniai:

1. Sudaryti sistemos naudotojo sąsajos prototipą;
2. Realizuoti saityno programą;
3. Realizuoti autentifikacijos ir paskyrų valdymo *API* programas;
4. Realizuoti tarpinio įgaliotojo serverio programą;
5. Realizuoti mobiliojo ryšio įrenginio taikomąją programą;
6. Sukonfigūruoti duomenų bazę ir sudaryti duomenų bazės schemą;
7. Sudaryti sistemos virtualią infrastruktūrą.

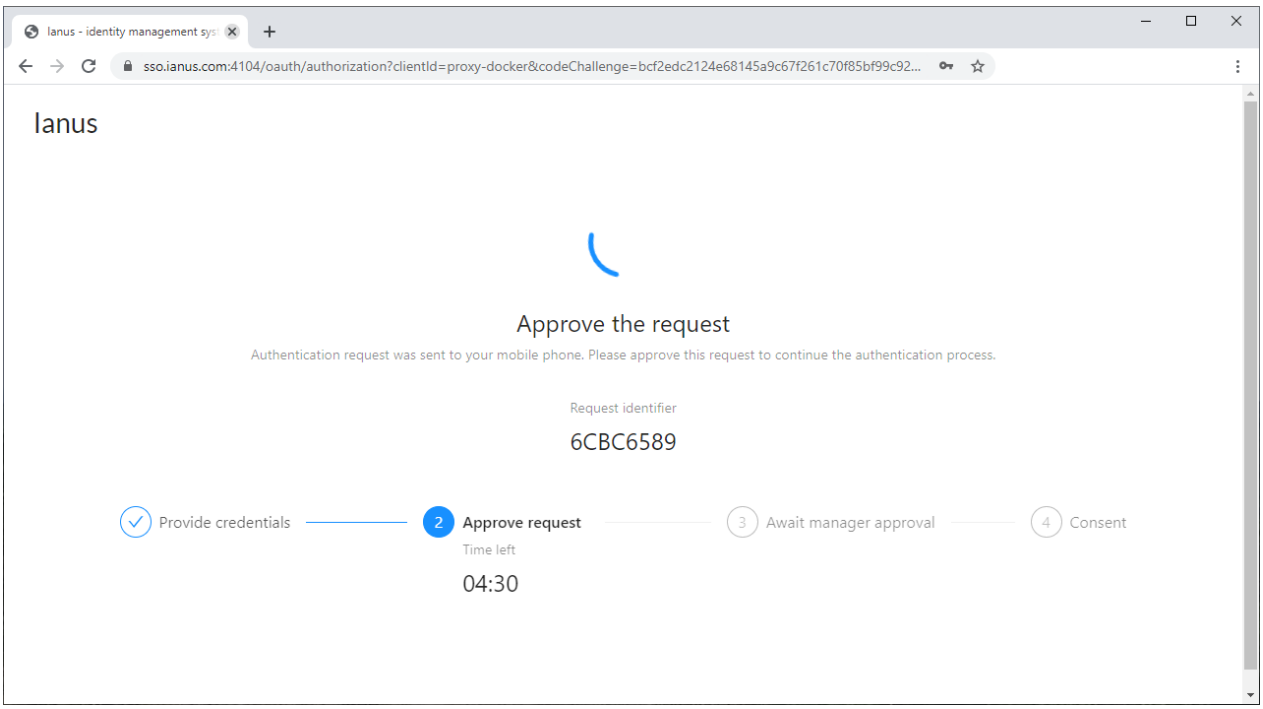
3.1. Naudotojo sąsajos prototipas

Naudotojo sąsajos projektavimo metu buvo sudaromi saityno programos ir mobiliojo ryšio įrenginio taikomosios programos langų prototipų eskizai. Eskizai buvo sudaromi pasitelkiant *Balsamiq Mockups v3.5.16* programinę įrangą. Eskizais buvo remiamasi naudotojo sąsajos realizavimo metu, pritaikant 5.8 nefunkciniame reikalavime nurodytą *Ant* dizaino sistemą ir jos komponentų biblioteką. Pirmiausiai yra pateikiamos realizuoto saityno programos prototipo iliustracijos.



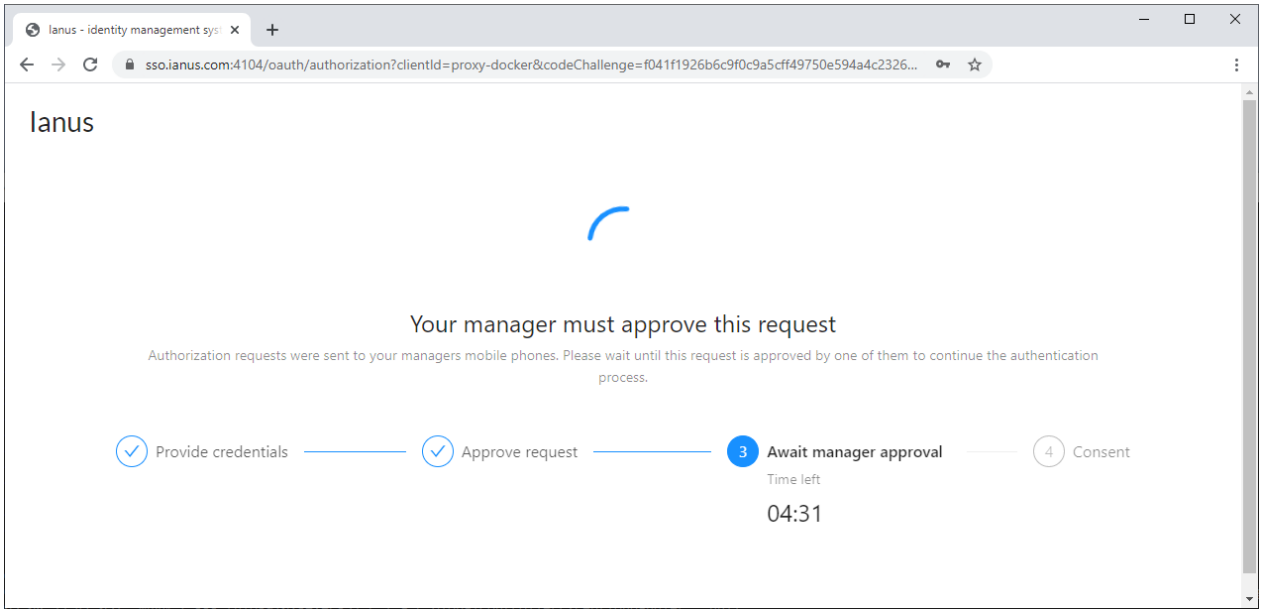
43 pav. Prisijungimo lango pirmojo žingsnio iliustracija

Prisijungimo lango pirmajame žingsnyje naudotojas, norėdamas pradėti autentifikacijos procesą, turi pateikti paskyros vardą ir slaptažodį. Pateikus teisingus duomenis, naudotojas patenka į antrąjį žingsnį.



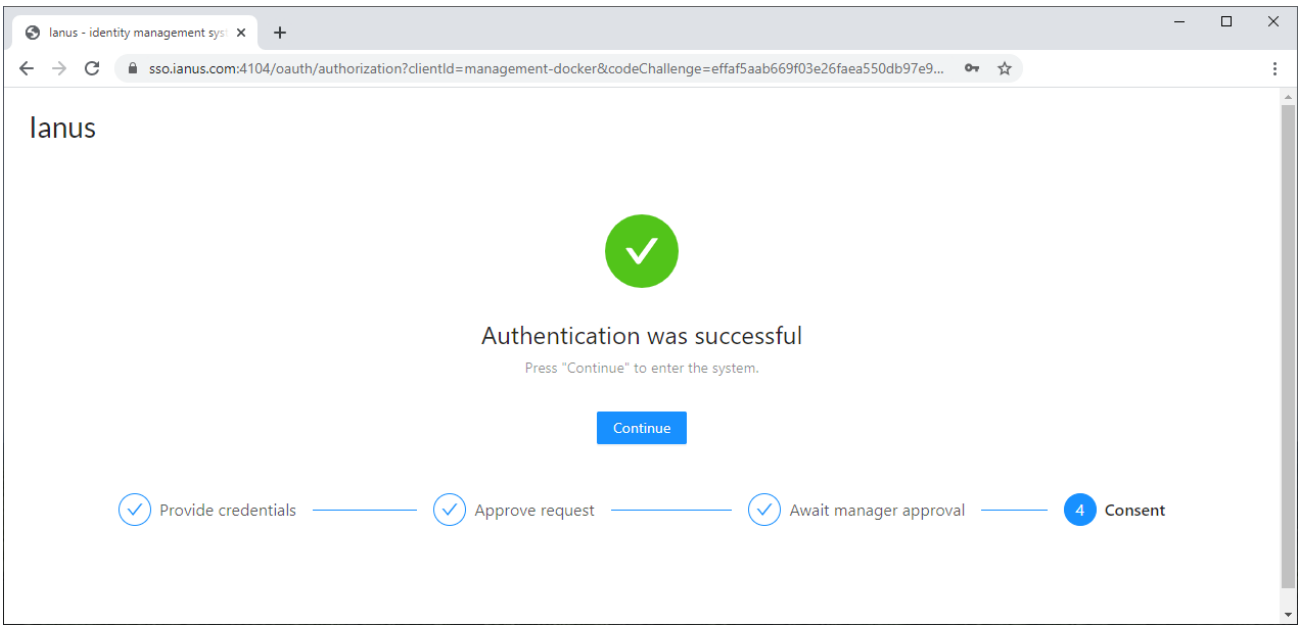
44 pav. Prisijungimo lango antrojo žingsnio iliustracija

Antrajame žingsnyje naudotojas yra informuojamas, jog autentifikacijos užklausa buvo nusiųsta į jo susietą mobiliojo ryšio įrenginį. Lange yra matomas laikas, skirtas užbaigti šį žingsnį, ir užklausos identifikatorius, kuris taip pat bus pateikiamas autentifikacijos užklausoje, pateikiamoje mobiliojo ryšio įrenginyje.



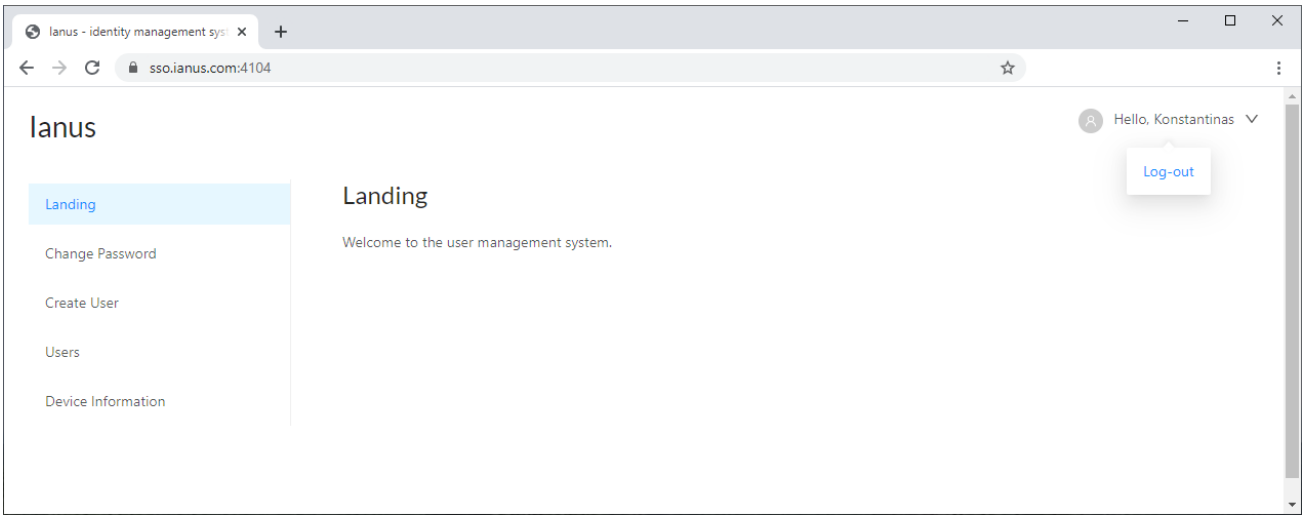
45 pav. Prisijungimo lango trečiojo žingsnio iliustracija

Trečiajame žingsnyje naudotojas yra informuojamas, jog autorizacijos užklausos buvo nusiųstos jo vadovams. Lange taip pat yra matomas laikas, skirtas užbaigti šį žingsnį.



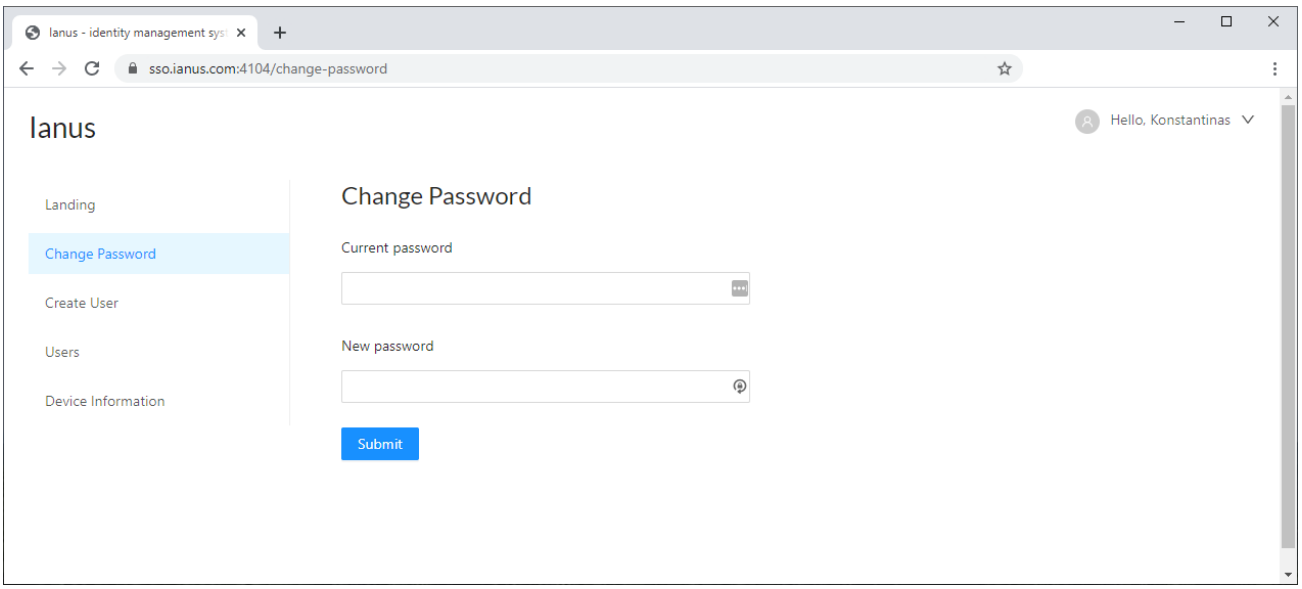
46 pav. Prisijungimo lango ketvirtojo žingsnio iliustracija

Ketvirtajame žingsnyje naudotojas yra informuojamas, jog autentifikacijos procesas buvo užbaigtas sėkmingai. Tuomet naudotojas gali sėkmingai pradėti naudotis sistema, paspausdamas mygtuką *Continue*.



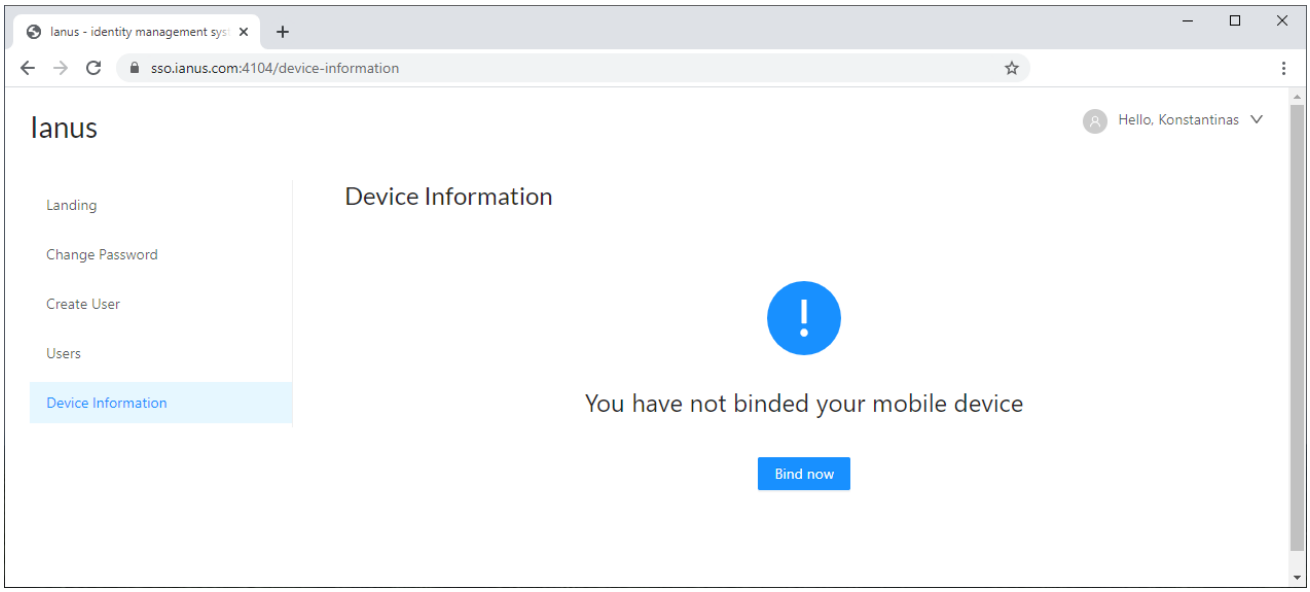
47 pav. Pagrindinio lango, matomo prisijungus prie sistemos, iliustracija

Pagrindinio lango, matomo prisijungus prie sistemos, iliustracijoje yra matoma, jog saityno programos langai yra sudaryti iš dviejų dedamųjų: navigacijos ir lango turinio. Navigacijoje yra pateikiamos nuorodos į kitus langus. Taip pat lango dešinėje, viršuje, yra nuoroda, pasveikinanti prisijungusį naudotoją. Paspaudus šią nuorodą, yra atidaromas kontekstinis meniu, kuriame yra pateikiama atsijungimo nuo sistemos nuoroda.



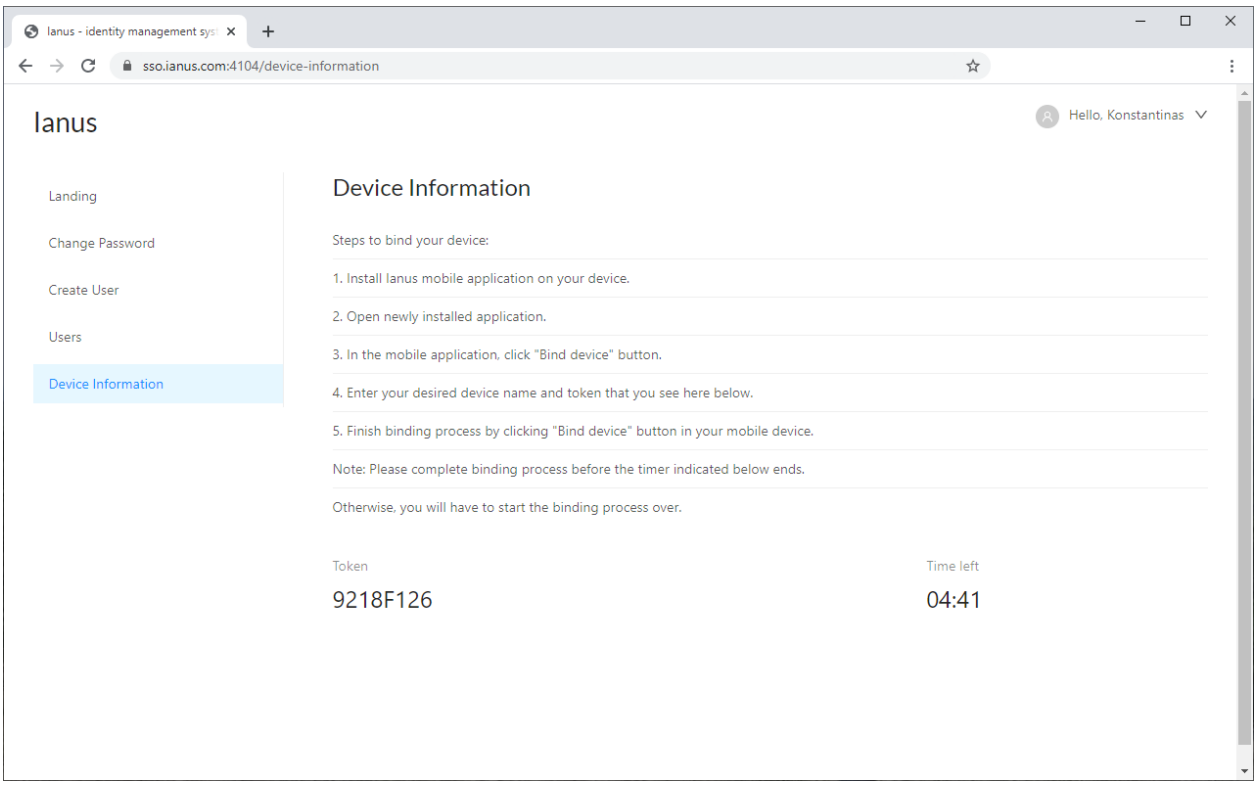
48 pav. Slaptažodžio keitimo lango iliustracija

Slaptažodžio keitimo lange yra pateikiama forma, kurioje naudotojas, norėdamas pasikeisti slaptažodį, turi pateikti dabartinį ir naują slaptažodžius. Slaptažodžio pakeitimas yra atliekamas paspaudus mygtuką *Submit*.



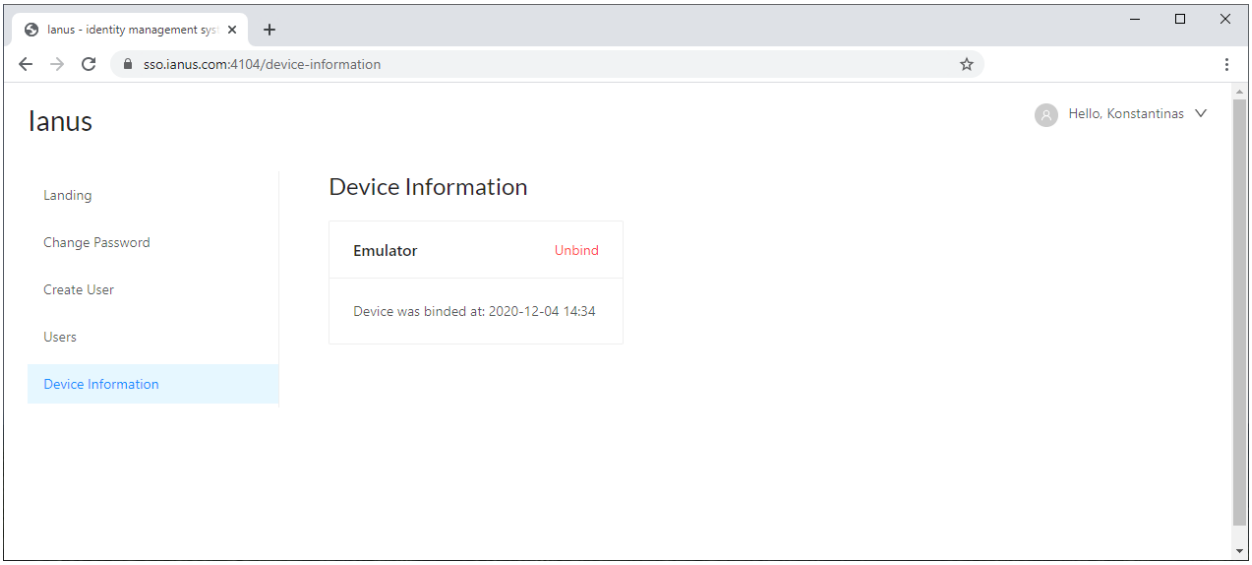
49 pav. Mobiliojo ryšio įrenginio informacijos lango, kuomet nėra susieto įrenginio, iliustracija

49 pav. yra pateikta iliustracija, nurodanti, kaip atrodo mobiliojo ryšio įrenginio informacijos langas, kuomet naudotojas nėra susiejęs savo mobiliojo ryšio įrenginio. Įrenginio susiejimo procedūra pradedama paspaudus mygtuką *Bind now*.



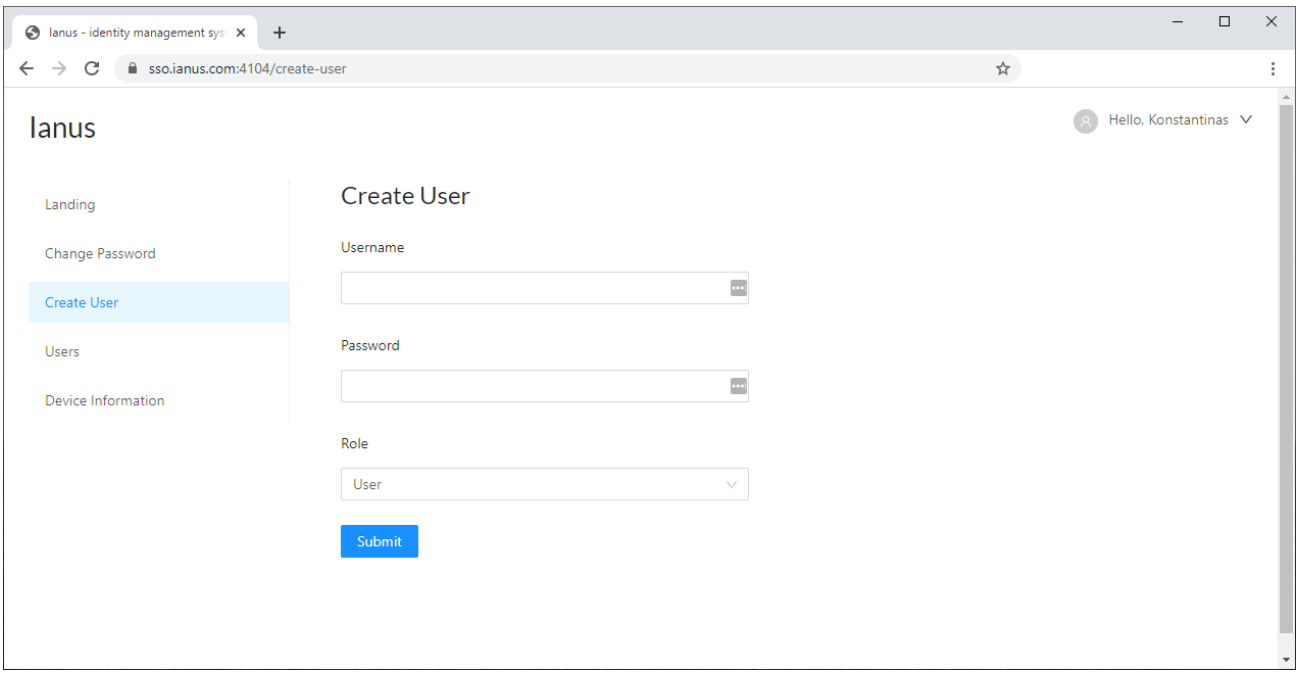
50 pav. Įrenginio susiejimo instrukcijų lango iliustracija

Įrenginio susiejimo instrukcijų lange yra nurodomos instrukcijos, skirtos atlikti mobiliojo ryšio įrenginio susiejimą ir taip jį pradėti naudoti patvirtinant autentifikacijos ir autorizacijos užklausas. Lange yra pateikiamas žetonas, kurį reikia įvesti mobiliojo ryšio įrenginyje ir laikas, likęs susiejimo proceso užbaigimui.



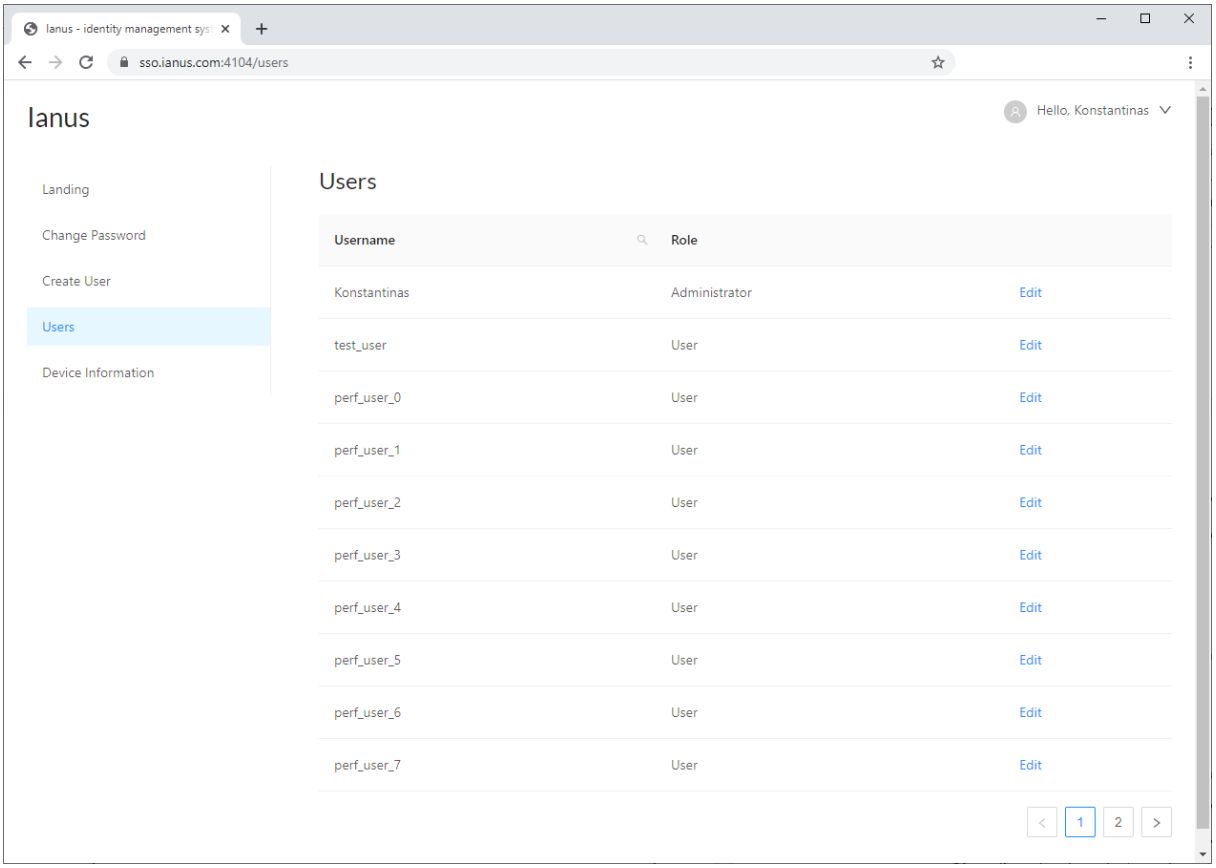
51 pav. Susieto įrenginio informacijos lango iliustracija

51 pav. yra pateikta iliustracija, nurodanti, kaip atrodo mobiliojo ryšio įrenginio informacijos langas, kuomet naudotojas yra susiejęs savo mobiliojo ryšio įrenginį. Lange yra matoma data, nurodanti kada įrenginys buvo susietas, bei įrenginio vardas. Lange taip pat yra pateikiama *Unbind* nuoroda, skirta atšaukti mobiliojo ryšio įrenginio susiejimą.



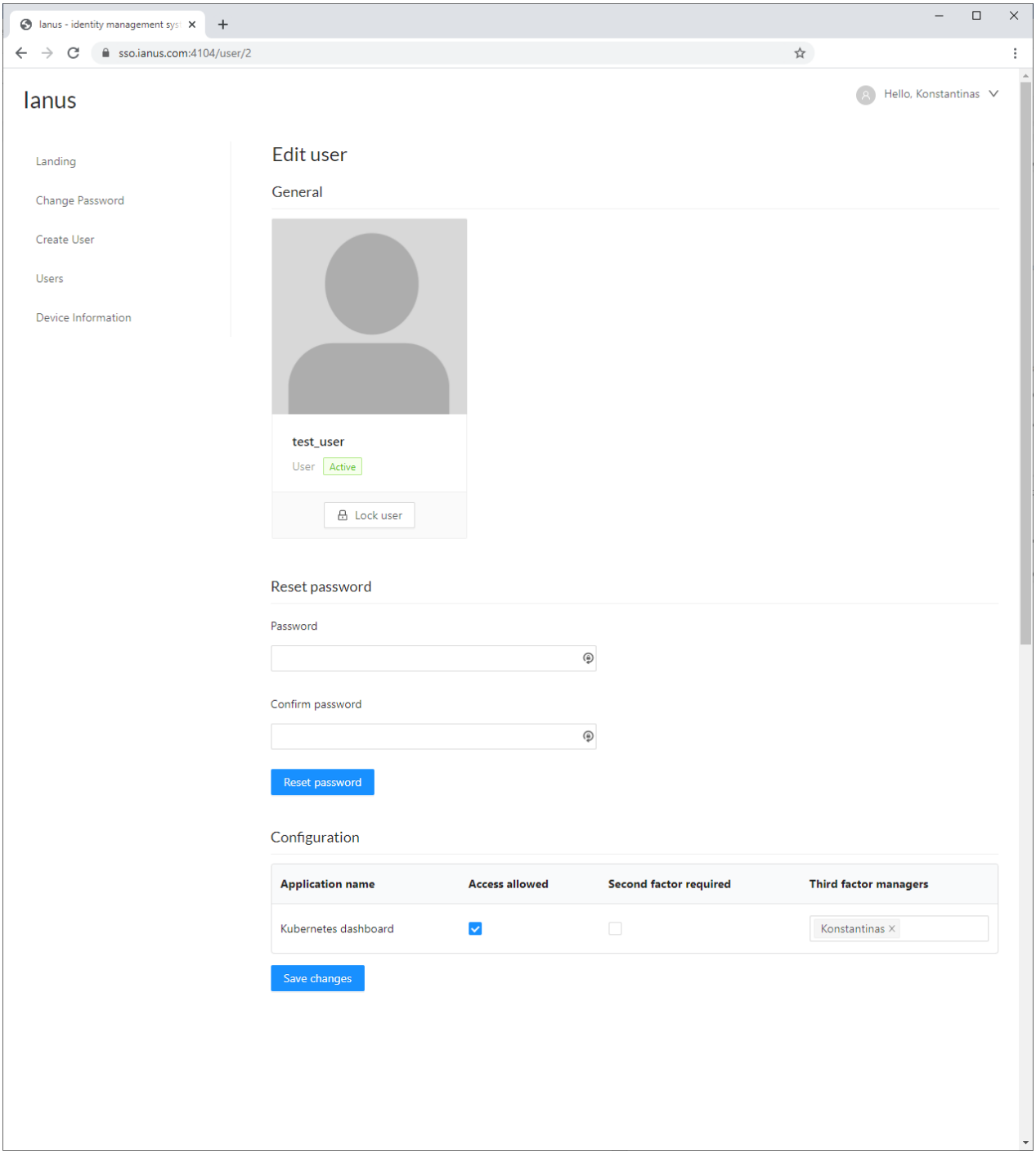
52 pav. Paskyros kūrimo lango iliustracija

Paskyros kūrimo lange yra pateikiama forma, kurioje naudotojas, norėdamas sukurti naują paskyrą, turi pateikti paskyros vardą, numatytąjį slaptažodį bei paskyros rolę. Šis langas yra prieinamas tik sistemos administratoriams. Paskyros sukūrimas yra atliekamas paspaudus mygtuką *Submit*.



53 pav. Paskyrų sąrašo lango iliustracija

Paskyrų sąrašo lange yra pateikiamas visų sistemos paskyrų sąrašas, kuris gali būti puslapiuojamas ir filtruojamas pagal paskyros vardą. Šis langas yra prieinamas tik sistemos administratoriams. Lentelės paskutiniame stulpelyje yra pateikiama *Edit* nuoroda, kurią paspaudus yra patenkama į paskyros informacijos ir konfigūracijos redagavimo langą, kuris yra pateikiamas 54 pav.

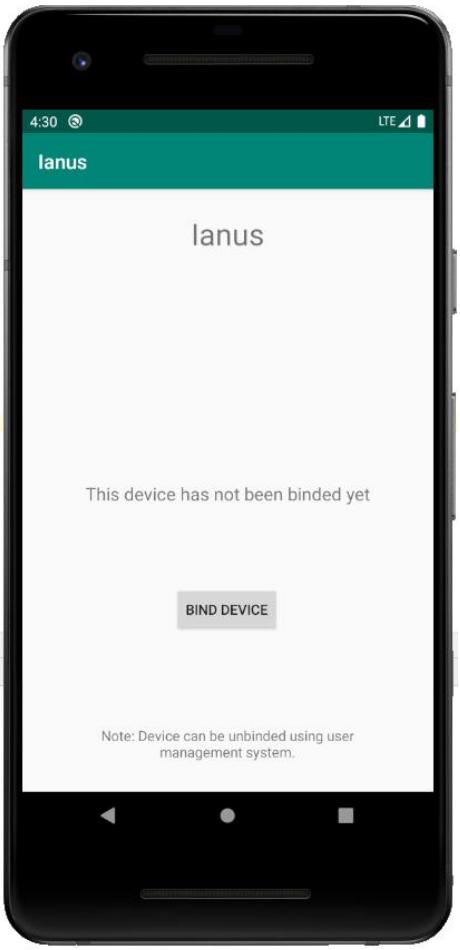


54 pav. Paskyros informacijos ir konfigūracijos redagavimo lango iliustracija

Paskyros informacijos ir konfigūracijos redagavimo lange yra pateikiama forma, kurioje galima atblokuoti arba užblokuoti paskyrą, pakeisti paskyros slaptažodį bei paskyros autentifikacijos ir autorizacijos nustatymus skirtingoms sistemoms. Šis langas taip pat yra prieinamas tik sistemos administratoriams. Konfigūracijos sekcijoje galima nurodyti, ar paskyrai jungiantis prie tam tikros sistemos turi būti taikomas antro faktoriaus (užklausos patvirtinimas naudojant susietą mobiliojo

ryšio įrenginį) panaudojimas ir ar turi būti taikomas trečiojo faktoriaus (užklausos, kurią patvirtina vadovas, naudodamas susietą mobiliojo ryšio įrenginį) panaudojimas ir kas bus nurodyti kaip paskyros vadovai. Paskyros konfigūracijos nustatymai yra išsaugomi paspaudus mygtuką *Save changes*.

Toliau yra pateikiamos realizuoto taikomosios mobiliojo ryšio įrenginio programos prototipo iliustracijos.



55 pav. Pagrindinio lango iliustracija, kuomet įrenginys nėra susietas

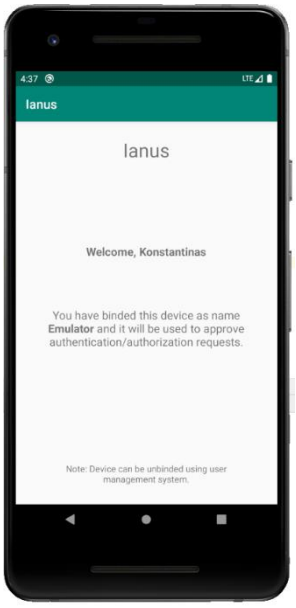
55 pav. yra pateikta iliustracija, kuri nurodo, kaip atrodo pagrindinis mobiliosios programos langas, kuomet įrenginys nėra susietas su naudotojo paskyra. Įrenginio susiejimo procesas yra vykdomas paspaudus mygtuką *Bind device*.

56 pav. Įrenginio susiejimo lango iliustracija

Įrenginio susiejimo lange yra matoma forma, kurioje naudotojas turi pateikti:

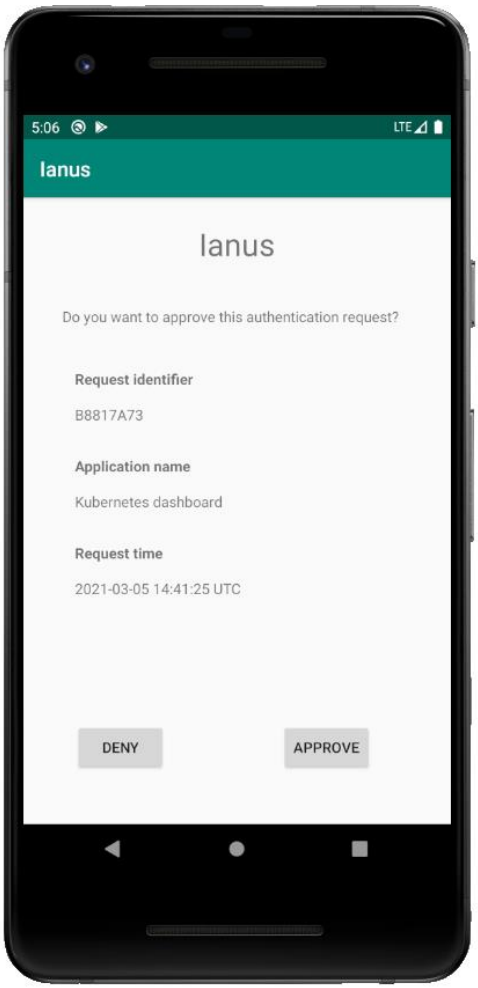
- Įrenginio vardą, kuris bus matomas saityno programoje;
- Žetoną, kuris yra pateikiamas saityno programoje, naudotojui pradedant vykdyti susiejimo procesą.

Įrenginio susiejimas yra užbaigiamas paspaudus mygtuką *Bind device*.



57 pav. Pagrindinio lango iliustracija, kuomet įrenginys yra susietas

57 pav. yra pateikta iliustracija, kuri nurodo, kaip atrodo pagrindinis mobiliosios programos langas, kuomet įrenginys yra susietas su naudotojo paskyra. Lange yra pateikiamas susietos paskyros vardas bei įrenginio vardas, kuris buvo nurodytas pildant įrenginio susiejimo formą. Apačioje taip pat yra pateikiamas pranešimas, nurodantis, kaip galima atlikti įrenginio susiejimo atšaukimo veiksmą.

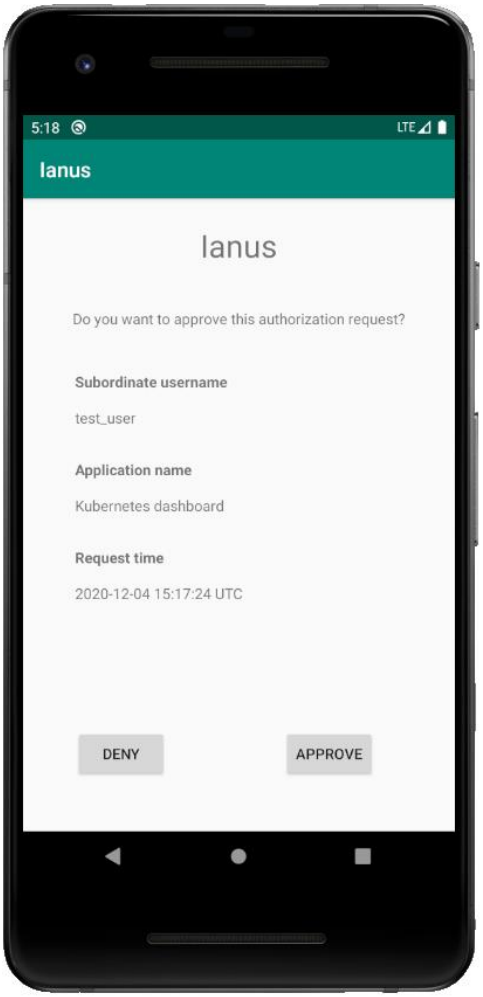


58 pav. Autentifikacijos užklausos lango iliustracija

Autentifikacijos užklausos langas naudotojui yra pateikiamas vykdant antrąjį autentifikacijos žingsnį. Šiame lange naudotojas mato:

- Užklausos identifikatorių, kuris taip pat yra pateikiamas saityno programoje;
- Sistemos, prie kurios yra jungiamasi, pavadinimą;
- Užklausos datą ir laiką.

Norėdamas atšaukti užklausą, naudotojas turi paspausti mygtuką *Deny*. Norėdamas patvirtinti užklausą, naudotojas turi paspausti mygtuką *Approve*. Jei naudotojas mobiliojo ryšio įrenginyje yra sukonfigūravęs *PIN* kodo ar piršto antspaudo panaudojimą, paspaudus veiksmo mygtuką, sistema paprašo pateikti vieną iš šių duomenų užklausos patvirtinimui. *PIN* kodo ar piršto antspaudo įvedimo lango dizainas priklauso nuo naudojamo įrenginio.



59 pav. Autorizacijos užklausos lango iliustracija

Autorizacijos užklausos langas yra pateikiamas vadovui, kuomet yra vykdomas trečiasis autentifikacijos žingsnis. Šiame lange vadovas mato:

- Pavaldinio paskyros vardą;
- Sistemos, prie kurios pavaldinys nori prisijungti, pavadinimą;
- Užklausos datą ir laiką.

Norėdamas atšaukti užklausą, vadovas turi paspausti mygtuką *Deny*. Norėdamas patvirtinti užklausą, vadovas turi paspausti mygtuką *Approve*. Jei vadovas mobiliojo ryšio įrenginyje yra sukonfigūravęs *PIN* kodo ar piršto antspaudo panaudojimą, paspaudus veiksmo mygtuką, sistema paprašo pateikti vieną iš šių duomenų užklausos patvirtinimui. *PIN* kodo ar piršto antspaudo pateikimo lango dizainas priklauso nuo naudojamo įrenginio.

3.2. Saityno programos realizacija

Saityno programos kūrimui buvo naudojama *TypeScript* programavimo kalba (3.8.3 versija) bei *HTML5* žymėjimo ir *CSS3* (naudojant *styled-components* biblioteką) stilių aprašymo kalbos. Taip pat buvo naudojama *React* 16.12.0 biblioteka.

Programos kūrimo metu lokalioje aplinkoje buvo naudojamas *webpack-dev-server* serveris, teikiantis saityno programos artefaktus. Serverio programa buvo paleidžiama naudojant *Node.js* 12.18.3 *LTS* versiją.

Produkcinės aplinkos artefaktų kūrimui buvo naudojama dedikuota *Webpack* konfigūracija, kuri sukompiliuoja, optimizuoja ir minimizuoja išeities kodą. Sugeneruoti artefaktai buvo diegiami į *NGINX* 1.19.2 serverį.

Programos realizacijos metu buvo panaudotos šios pagalbinės bibliotekos:

- *antd* – skirta *Ant* dizaino bibliotekos komponentų panaudojimui;
- *styled-components* – skirta *CSS3* stilių pritaikymui *React* komponentams;
- *formik* – skirta palengvinti formų kūrimą;
- *axios* – skirta vykdyti *HTTP* užklausas į *API* programų prieigos taškus;
- *zxcvbn* – skirta slaptažodžių sudėtingumo tikrinimui;
- *lodash* – skirta pagalbinių funkcijų panaudojimui.

Realizacijos metu buvo sukurti 20 pakartotinai naudojamų ir 14 konkretiems puslapiams skirtų *React* komponentų.

Norint paleisti programą virtualioje infrastruktūroje, reikia sukurti *Docker* atvaizdą. Tai galima atlikti šakniniame projekto aplankale įvykdžius komandą (kur 1.0.0 yra norima programos versija):

```
docker build --tag ianus/ianus-frontend:1.0.0 .
```

3.3. *API* programų realizacija

API programos buvo programuojamos naudojant *Java* programavimo kalbą (*OpenJDK* 13.0.2 versiją) bei *Spring* 5.2 karkasą. Programavimo darbai buvo atliekami naudojant *IntelliJ IDEA* 2019.3.2 integruotą kūrimo aplinką. Programų kūrimui taip pat buvo naudojamas *Gradle* 6.3 įrankis.

API programų kodo repositorijoje buvo sukurti 4 moduliai:

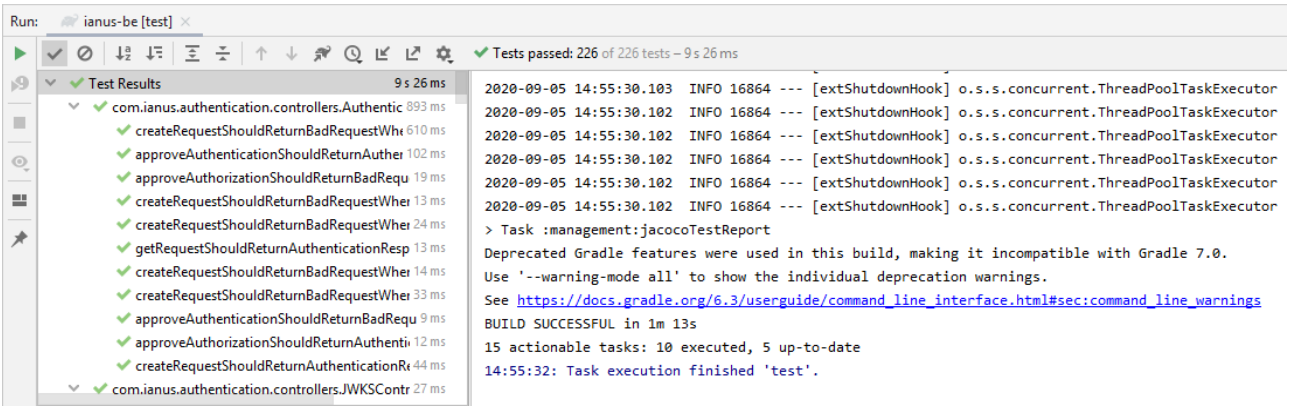
- *authentication* – kuris realizuoja autentifikacijos *API* programą;
- *management* – kuris realizuoja paskyrų valdymo *API* programą;
- *common* – kuriame yra talpinamos bendros klasės, kurias naudoja kiti moduliai;
- *firebase-proxy* – kuriame yra talpinamas kodas, susijęs su *Firebase API* integracija.

Programų realizacijos metu buvo panaudotos šios pagalbinės bibliotekos:

- *jackson* – skirta serializuoti/deserializuoti duomenis į *JSON* formatą;
- *jjwt* – skirta *JWT* žetonų sukūrimui, pasirašymui ir verifikavimui;
- *spring-security-oauth2-jose* – kuri suteikė galimybę apsaugoti paskyrų valdymo *API* programą kaip *OAuth 2.0* resursų serverį, panaudojant *JWT* žetonus;

– *lombok* – skirta automatiškai sugeneruoti *Java* klasių atributų prieigos/modifikavimo metodus bei konstruktorius;
– *zxcvbn4j* – skirta slaptažodžių sudėtingumo tikrinimui;
– *postgresql* – skirta prisijungimui prie *PostgreSQL* duomenų bazės per *JDBC* tvarkyklę.

API programų testavimui buvo naudojama *JUnit 5* biblioteka. Iš viso buvo parašyti 226 vienetų testai, kurie buvo skirti ištestuoti didžiąją dalį *API* kodo. 60 pav. yra pateikti vienetų testų vykdymo rezultatai, kuomet buvo užbaigta programų realizacija.



60 pav. *API* programų vienetų testų vykdymo rezultatai

Norint paleisti *API* programas virtualioje infrastruktūroje, reikia sukurti du *Docker* atvaizdus. Tai galima atlikti šakniniame projekto aplankale įvykdžius komandas (kur 1.0.0 yra norima programos versija):

docker build -t ianus/authentication-service:1.0.0 -f authentication/Dockerfile .
docker build -t ianus/management-service:1.0.0 -f management/Dockerfile .

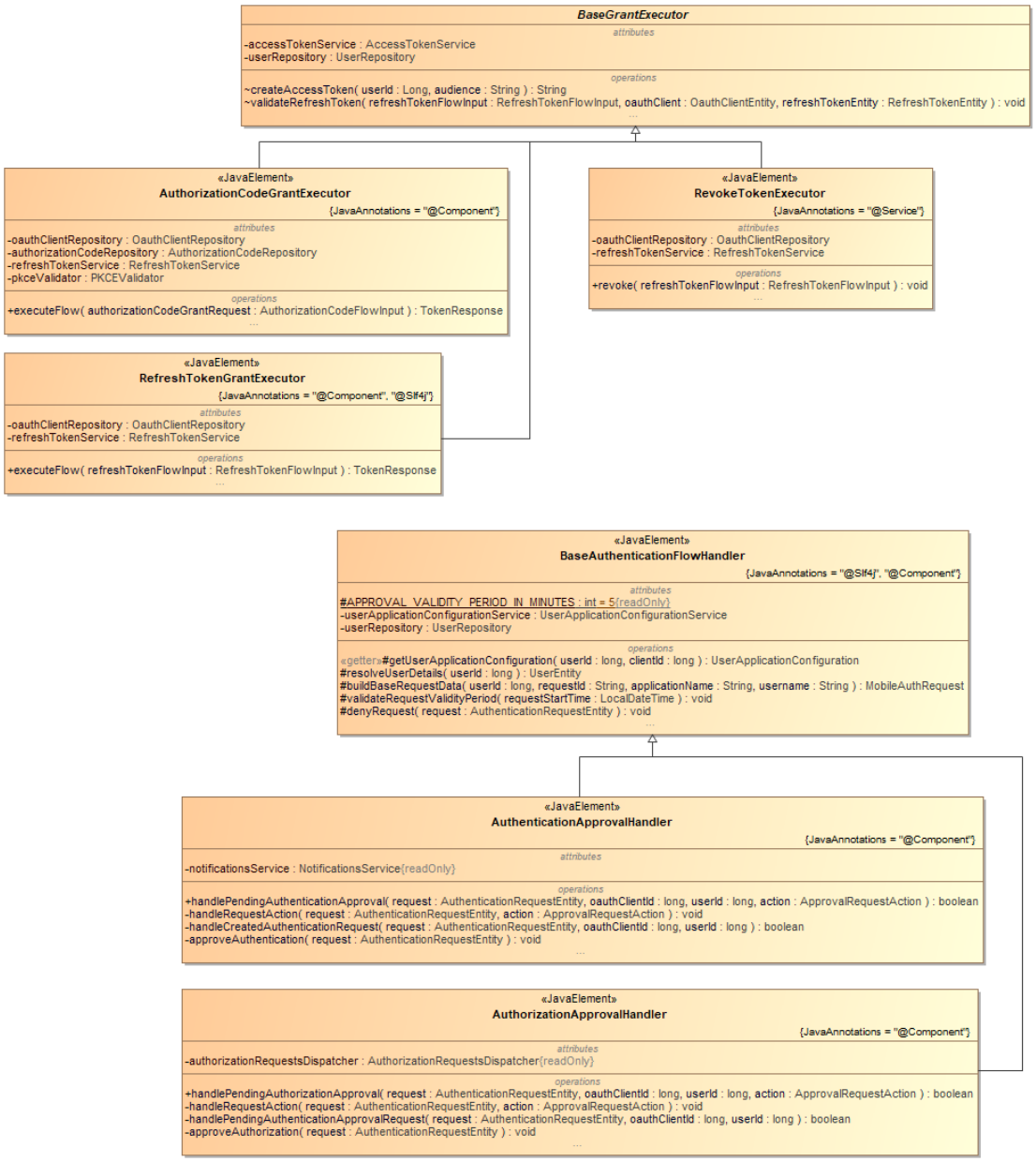
3.3.1. Autentifikacijos *API* klasių diagramos

Žemiau yra pateikiamos autentifikacijos *API* programos klasių diagramos. 61 pav. yra pateikiama valdiklių klasių diagrama. Valdiklių klasės yra skirtos priimti *HTTP* užklausas, ateinančias į prieigos taškus (angl. *endpoints*), patikrinti, ar užklausos kūnas bei parametrai yra teisingi, ir perduoti užklausos vykdymą servisų klasėms. Iš viso autentifikacijos *API* programoje buvo sukurtos 4 valdiklių klasės ir 8 prieigos taškai.

61 pav. Autentifikacijos *API* programos valdiklių klasių diagrama

62 pav. yra pateikiama servisų klasių diagrama. Šios klasės yra atsakingos už dalykinės srities bei funkcinių reikalavimų realizaciją. Iš viso autentifikacijos *API* programoje buvo sukurta 19 servisų klasių.

62 pav. Autentifikacijos *API* programos servisų klasių diagrama



63 pav. Autentifikacijos *API* programos autentifikacijos ir *OAuth 2.0* užklausų apdorojimo klasių diagrama

63 pav. yra pateikiama autentifikacijos ir *OAuth 2.0* užklausų apdorojimo klasių diagrama. *AuthenticationApprovalHandler* ir *AuthorizationApprovalHandler* klasės yra atsakingos už autentifikacijos bei autorizacijos užklausų apdorojimo logiką. Kadangi dalis logikos yra naudojama abiejų tipų užklausose, bendra logika buvo iškelta į bazinę *BaseAuthenticationFlowHandler* klasę.

Šioje diagramoje taip pat yra pateikiamos ir *OAuth 2.0* užklausų apdorojimo klasės, kurios yra skirtos apdoroti šias *OAuth 2.0* protokolo užklausų rūšis (angl. *flows*):

- Žetono atšaukimo (angl. *token revocation*);
- Autorizacijos kodo (angl. *authorization code*);
- Naujo žetono išdavimo (angl. *refresh token*).

Dalis užklausų apdorojimo logikos taip pat yra pakartotinai panaudojama, tad bendra logika buvo iškelta į bazinę *BaseGrantExecutor* klasę.

3.3.2. Paskyrų valdymo *API* klasių diagramos

Žemiau yra pateikiamos paskyrų valdymo *API* programos klasių diagramos. 64 pav. yra pateikiama valdiklių klasių diagrama. Valdiklių klasės yra skirtos priimti *HTTP* užklausas, ateinančias į prieigos taškus (angl. *endpoints*), patikrinti, ar užklausos kūnas bei parametrai yra teisingi, ir perduoti užklausos vykdymą servisų klasėms. Iš viso paskyrų valdymo *API* programoje buvo sukurtos 7 valdiklių klasės ir 19 prieigos taškų.

64 pav. Paskyrų valdymo *API* programos valdiklių klasių diagrama

65 pav. yra pateikiama servisų klasių diagrama. Šios klasės yra atsakingos už dalykinės srities bei funkcinių reikalavimų realizaciją. Iš viso paskyrų valdymo *API* programoje buvo sukurta 10 servisų klasių.

65 pav. Paskyrų valdymo *API* programos servisų klasių diagrama

3.4. Tarpinio įgaliotojo serverio programos realizacija

Tarpinio įgaliotojo serverio programos kūrimui buvo naudojama *JavaScript* programavimo kalba (*ECMAScript* 10 standartas) bei *Express* karkasas (4.17.1 versija). Serverio programa buvo paleidžiama naudojant *Node.js* 12.18.3 *LTS* versiją ir *nodemon* foninį procesą (angl. *daemon*).

Programos realizacijos metu buvo panaudotos šios pagalbinės bibliotekos:

– *jsonwebtoken* – skirta *JWT* žetonų verifikavimui;
– *jwks-rsa* – skirta gauti asimetrinį šifravimo raktą iš autentifikacijos *API* programos, kuriuo yra pasirašomi *JWT* žetonai;
– *ejs* – skirta panaudoti *HTML* puslapių šablonus *Express* programoje.

Programos testavimui buvo naudojama *jest* biblioteka. Iš viso buvo parašyti 17 vienetų testų, kurie buvo skirti ištestuoti *HTTP* valdiklius bei dalykinę logiką. 66 pav. yra pateikti vienetų testų vykdymo rezultatai, kuomet buvo užbaigta programos realizacija.

```
PS F:\ianus\ianus-proxy> npm run test

> ianus-proxy@1.0.0 test F:\ianus\ianus-proxy
> jest --config jest.config.js

 PASS  handlers/specs/authorize.spec.js (6.02 s)
[2020-09-05 09:49:49.906 +0300] INFO  (8616 on          ): Executing authorization handler
[2020-09-05 09:49:49.911 +0300] INFO  (8616 on          ): Redirecting to identity manager
[2020-09-05 09:49:49.909 +0300] INFO  (9804 on          ): Executing sign-out handler
[2020-09-05 09:49:49.910 +0300] INFO  (16592 on         ): Executing callback handler
[2020-09-05 09:49:49.913 +0300] GET /authorize 302
 PASS  handlers/specs/sign-out.spec.js (6.101 s)
 PASS  handlers/specs/callback.spec.js (6.105 s)
[2020-09-05 09:49:49.986 +0300] GET /callback?state=test2 200
[2020-09-05 09:49:49.996 +0300] INFO  (16592 on          ): Executing callback handler
[2020-09-05 09:49:49.997 +0300] WARN  (16592 on          ): Invalid state parameter provided
[2020-09-05 09:49:49.999 +0300] GET /callback?state=bad-state 200
[2020-09-05 09:49:50.004 +0300] INFO  (16592 on          ): Executing callback handler
[2020-09-05 09:49:50.005 +0300] ERROR (16592 on          ): Failed to exchange token. Error: [object Object]
[2020-09-05 09:49:50.007 +0300] GET /callback?state=test2 200
[2020-09-05 09:49:49.986 +0300] GET /sign-out 200
[2020-09-05 09:49:49.996 +0300] INFO  (9804 on          ): Executing sign-out handler
[2020-09-05 09:49:49.998 +0300] GET /sign-out 200
 PASS  handlers/specs/resource.spec.js
 PASS  handlers/specs/sessionStatusHandler.spec.js
[2020-09-05 09:49:50.515 +0300] INFO  (16592 on         ): Executing session status handler
[2020-09-05 09:49:50.520 +0300] GET /api/v1/login/status 200
[2020-09-05 09:49:50.515 +0300] INFO  (9804 on          ): Starting token refresh action
[2020-09-05 09:49:50.522 +0300] INFO  (9804 on          ): retrieving protected resource
[2020-09-05 09:49:50.524 +0300] GET /resource 200
[2020-09-05 09:49:50.531 +0300] INFO  (9804 on          ): Starting token refresh action
[2020-09-05 09:49:50.531 +0300] INFO  (9804 on          ): Refresh token has expired
[2020-09-05 09:49:50.532 +0300] GET /resource 401
[2020-09-05 09:49:50.545 +0300] INFO  (16592 on         ): Executing session status handler
[2020-09-05 09:49:50.546 +0300] GET /api/v1/login/status 200
 PASS  handlers/specs/home.spec.js
[2020-09-05 09:49:50.616 +0300] GET /home 200
 PASS  domain/specs/accessTokenValidator.spec.js

Test Suites: 7 passed, 7 total
Tests:       17 passed, 17 total
Snapshots:   0 total
Time:        8.096 s
Ran all test suites.
```

66 pav. Tarpinio įgaliotojo serverio vienetų testų vykdymo rezultatai

Norint paleisti programą virtualioje infrastruktūroje, reikia sukurti *Docker* atvaizdą. Tai galima atlikti šakniniame projekto aplankale įvykdžius komandą (kur 1.0.0 yra norima programos versija):

```
docker build --tag ianus/ianus-proxy:1.0.0 .
```

3.5. Mobiliojo ryšio įrenginio taikomosios programos realizacija

Mobiliojo ryšio įrenginio taikomosios programos kūrimui buvo naudojama *Kotlin* programavimo kalba (1.3.61 versija). Programavimo darbai buvo atliekami naudojant *Android Studio* 3.5.1 integruotą kūrimo aplinką. Lokalioje aplinkoje, mobilioji programa buvo paleidžiama naudojant *Pixel 2 API 29* virtualų įrenginį.

Programos realizacijos metu buvo panaudotos šios pagalbinės bibliotekos:

- *jjwt* – skirta *JWT* žetonų sukūrimui ir pasirašymui;
- *firebase-messaging* – skirta integracijai su *Firebase* paslauga, per kurią yra siunčiami pranešimai iš autentifikacijos *API* programos;
- *biometric* – skirta panaudoti *Android* sistemos teikiamą *PIN* kodo ir pirštų antspaudų funkcionalumą.

67 pav. yra pateikta mobiliojo ryšio įrenginio taikomosios programos klasių diagrama. Šioje diagramoje yra pavaizduotos dviejų tipų klasės: *Activity* tipo klasės, kurios valdo tam tikro programos lango veikimą, ir *Service* tipo klasės, kurios yra atsakingos už tam tikros dalykinės logikos realizaciją.

67 pav. Mobiliojo ryšio įrenginio taikomosios programos klasių diagrama

3.6. Duomenų bazės konfigūracija

Sistemos duomenų saugojimui buvo pasirinkta reliacinė *PostgreSQL* duomenų bazė. Duomenų bazė buvo leidžiama *Docker* konteineryje naudojant oficialų *postgres:12.1* atvaizdą. Duomenų bazės valdymui ir duomenų skaitymui bei modifikavimui buvo naudojamas *DBeaver* 7.0.0 įrankis.

Norint paleisti duomenų bazės konteinerį, reikia pasinaudoti *docker-compose* įrankiu ir paleisti *yml* konfigūraciją, esančią *API* programų repositorijoje. Prieš paleidžiant konfigūraciją, reikia sukurti failą pavadinimu *.env* ir jame įrašyti eilutę „POSTGRES_PASSWORD=password“, kur „password“ yra norimas pradinis duomenų bazės slaptažodis. Tai atlikus, galima vykdyti komandą:

```
docker-compose -f devops/docker/postgresql.yml up -d
```

Ši komanda paleidžia *PostgreSQL* duomenų bazę, kuri klausosi susijungimų per 5432 *TCP* prievadą.

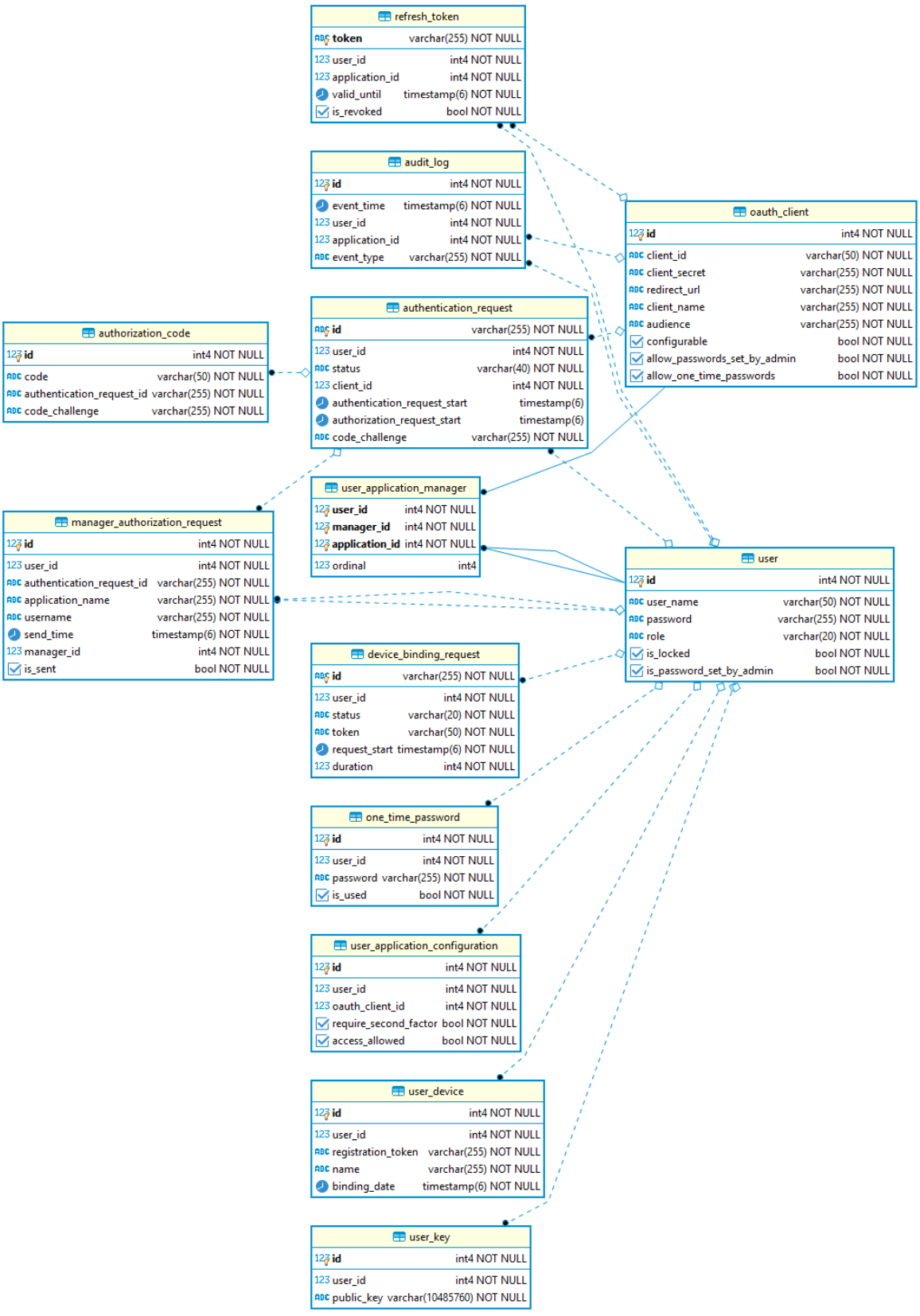
Paleidus duomenų bazės konteinerį, reikia sukurti naują duomenų bazės naudotoją. Tai galima atlikti prie *DBVS* prisijungus naudojant *DBeaver* įrankį. *DBVS* prisijungimo vardas yra *postgres*, o slaptažodis buvo nurodytas *.env* faile. Prisijungus prie *DBVS*, duomenų bazės naudotojas yra sukuriamas įvykdant šias *SQL* užklausas, kur *{PASSWORD}* yra norimas naudotojo slaptažodis:

```
CREATE DATABASE ianus;
CREATE USER ianus WITH ENCRYPTED PASSWORD '{PASSWORD}';
GRANT ALL PRIVILEGES ON DATABASE ianus TO ianus;
```

Sukūrus duomenų bazės naudotoją, reikia sukurti pačią duomenų bazės schemą ir ją užpildyti pradiniais duomenimis. Tam prie *DBVS* reikia prisijungti su naujai sukurtu naudotoju ir įvykdyti *SQL* užklausas, kurios yra saugomos *API* programų repositorijoje esančiuose failuose: *database/schema.sql* ir *database/data.sql*. Sėkmingai įvykdžius šias užklausas, duomenų bazė yra parengta naudojimui.

68 pav. yra pateikta duomenų bazės schemos diagrama, kurioje yra nurodomos duomenų bazėje esančios lentelės, jų atributai bei tarpusavio ryšiai. Pagrindinės lentelės yra:

- *user* – kurioje yra saugoma pagrindinė informacija apie naudotoją;
- *oauth_client* – kurioje yra saugoma pagrindinė informacija apie *OAuth 2.0* klientą, kuris integruojasi su vieningo prisijungimo sistema;
- *authentication_request* – kurioje yra saugoma informacija apie autentifikacijos užklausą;
- *user_application_configuration* – kurioje yra saugoma informacija apie naudotojo prieigą prie skirtingų *OAuth 2.0* klientų.



68 pav. Duomenų bazės schemos diagrama

3.7. Virtualios infrastruktūros paleidimas lokalioje aplinkoje

Norint paleisti visus sistemos komponentus lokalioje aplinkoje, reikia turėti *Docker* virtualizacijos įrankį. Taip pat prieš paleidžiant virtualią infrastruktūrą, reikia:

- Paleisti ir sukonfigūruoti duomenų bazę (poskyris „Duomenų bazės konfigūracija“);
- Sukurti visų sistemos komponentų *Docker* atvaizdus (angl. *images*);
- *API* programų repositorijos šakniniame kataloge *.env.example* failo pagrindu sukurti konfigūracijos failą. Šiame faile reikia nurodyti šiuos parametrus:
  - DATABASE_PASSWORD – *DBVS* naudotojo slaptažodis;
  - DATABASE_USERNAME - *DBVS* naudotojo vardas;
  - KEY_PASSPHRASE – *Java* raktų saugyklos slaptažodis;
  - KEY_FILE – *Java* raktų saugyklos failo vardas;
  - KEY_NAME – *Java* raktų saugykloje esančio rakto vardas;
  - FIREBASE_SERVER_KEY – *Firebase API* raktas;
  - OAUTH_CLIENT_ID – Paskyrų valdymo *API* programos *OAuth 2.0* kliento vardas;
  - OAUTH_CLIENT_SECRET - Paskyrų valdymo *API* programos *OAuth 2.0* kliento paslaptis;
  - DOCKER_HOST_IP – Tėvinės (angl. *host*) mašinos *IP* adresas, kuris yra matomas iš *Docker* konteinerio.

Atlikus šiuos žingsnius, virtuali infrastruktūra gali būti paleista iš šakninio *API* programų katalogo vykdant šią komandą:

```
docker-compose -f devops/docker/infrastructure.yml up -d
```

Apie sėkmingą infrastruktūros paleidimą yra pranešama terminale. Sėkmingo paleidimo pranešimo pavyzdys yra pateikiamas 69 pav.

```
F:\ianus\ianus-be>docker-compose -f devops/docker/infrastructure.yml up -d
WARNING: Found orphan containers (ianus-pg) for this project. If you removed or renamed
Creating ianus-management     ... done
Creating ianus-authentication ... done
Creating ianus-frontend       ... done
Creating ianus-proxy          ... done
Creating ianus-k8-dashboard   ... done
Creating ianus-k8-api-mocks   ... done
```

69 pav. Pranešimas apie sėkmingą infrastruktūros paleidimą

Virtuali infrastruktūra gali būti išjungta įvykdant šią komandą:

```
docker-compose -f devops/docker/infrastructure.yml down
```

3.8. Išvados

Realizacijos dalies metu buvo sukurtas sprendimo prototipas, pagrįstas vieningo prisijungimo sistema, taikančia suprojektuotą autentifikavimo metodą, ir tarpiniu įgaliotuoju serveriu, kuris naudoja vieningo prisijungimo sistemos funkcionalumą ir apsaugo nuotolinę prieigą prie *Kubernetes* valdymo skydelio.

Sprendimo prototipui buvo identifikuoti esminiai komponentai - saityno programa, dvi *API* programos, tarpinio įgaliotojo serverio programa, mobiliojo ryšio įrenginio taikomoji programa ir reliacinė duomenų bazė, kurios klientai yra dvi *API* programos, naudojančios tą pačią duomenų bazės schemą ir bendrus duomenis. Visų komponentų realizacijos metu buvo nustatyta, kad jie atitinka visus keliamus funkcinius ir nefunkcinius reikalavimus, tad sprendimo prototipas gali būti sėkmingai naudojamas kritinės infrastruktūros sistemų aplinkoje.

Nustatyta, kad prieigos prie kritinės infrastruktūros sistemų autorizavimas, mobiliojo ryšio įrenginio taikomosios programos funkcijų realizavimas bei naudotojo autentifikacijos/autorizacijos užklausų patvirtinimas ar atšaukimas bus tinkamiausiai realizuotas naudojant *JWT* žetonus, o jų pasirašymui yra naudojamas *RS512* algoritmas, taip užtikrinant saugų žetonų naudojimą.

Nustatyta, kad *API* bei tarpinio įgaliotojo serverio programoms parašyti vienetų testai leido aptikti ir ištaisyti ne vieną defektą sprendimo tobulinimo metu.

Realizacijos metu taip pat nustatyta, kad *Docker* virtualizacijos sprendimas, panaudotas visų sprendimo komponentų konteinerizavimui, leido greitai ir patogiai paleisti visą virtualią infrastruktūrą lokalioje aplinkoje.

Tinkamas sprendimo prototipo realizavimas sąlygoja tolimesnę sprendimo veikimo tiriamąją dalį.

4. Kelių faktorių autentifikacijos metodo ir jį palaikančios sistemos tyrimas

Tyrimo dalies tikslas – ištirti ir įvertinti siūlomo autentifikacijos metodo ir jį realizuojančios vieningo prisijungimo sistemos prototipo kokybines ir kiekybines charakteristikas. Šiam tikslui pasiekti yra keliami uždaviniai:

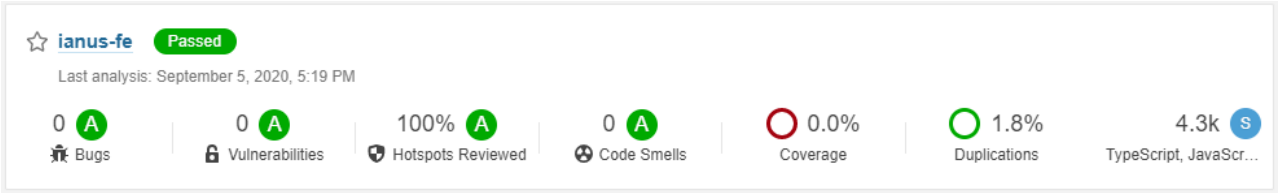
1. Atlikti prototipą sudarančių komponentų statinę kodo analizę;
2. Ištirti siūlomo metodo kokybines charakteristikas ir jas palyginti su kitų mokslininkų siūlomais metodais;
3. Ištirti serverių konfigūracijos saugumo aspektus vykdant dinaminę ir statinę konfigūracijos analizę;
4. Atlikti sistemos atsparumo nuo injekcijų ir pavojingų atakos vektorių tyrimą;
5. Nustatyti ir įvertinti sistemos prieigos taškų greitaveiką;
6. Ištirti ir įvertinti metodo konfigūracijos ir autentifikavimo procesų atlikimo greitį.

4.1. Sistemos komponentų statinė kodo analizė

Komponentų kodo kokybė buvo tikrinama vykdant statinę kodo analizę su *SonarQube* 8.4.2 įrankiu.

4.1.1. Saityno programos kodo analizė

Analizuojant saityno programą, įrankis iš viso išanalizavo 4325 kodo eilutes ir pateikė rezultatus, kurie yra pavaizduoti 70 pav.

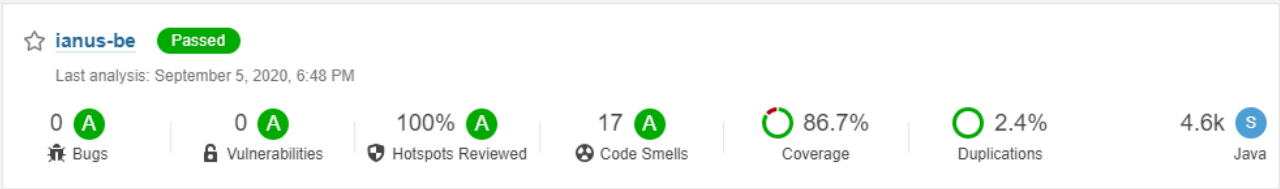


70 pav. Saityno programos statinės kodo analizės rezultatai

Vykdant analizę pirmą kartą, įrankis aptiko vieną saugumo klaidą, nurodydamas, jog yra naudojama potencialiai nesaugi *SHA256* maišos funkcija. Ši funkcija yra naudojama *PKCE* protokole sukuriant verifikavimo kodo santrauką. Pagal šio protokolo specifikaciją, *SHA256* yra tinkama ir pakankamai saugi funkcija, skirta sukurti reikiamų duomenų santrauką, tad aptikta saugumo klaida buvo pažymėta kaip išspręsta. Taip pat buvo aptikta viena klaida bei 35 netinkamo kodo vietos. Visos aptiktos klaidos ir netinkamo kodo vietos buvo ištaisytos.

4.1.2. *API* programų kodo analizė

Analizuojant *API* programas, įrankis iš viso išanalizavo 4635 kodo eilutes ir pateikė rezultatus, kurie yra pavaizduoti 71 pav. Rezultatuose matoma, jog buvo pasiektas 86.7% kodo padengimas vienetų testais ir buvo įgyvendintas nefunkcinis reikalavimas pasiekti 85% kodo padengimą.



71 pav. *API* programų statinės kodo analizės rezultatai

Vykdant analizę pirmą kartą, įrankis aptiko 14 saugumo klaidų bei 136 netinkamo kodo vietas. Didžioji dalis netinkamo kodo vietų buvo ištaisytos. Po pataisymų įrankis aptiko likusias 17 netinkamo kodo vietų, kurių ištaisymas yra mažo prioriteto.

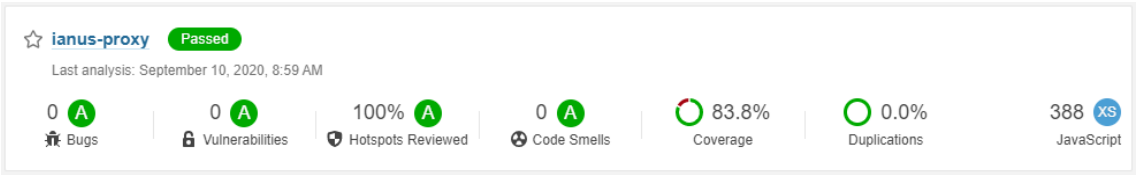
Visos 14 saugumo klaidų buvo peržiūrėtos. Aukšto prioriteto klaida buvo *Spring Security CSRF* apsaugos išjungimas paskyrų valdymo *API* programoje. Šis funkcionalumas buvo išjungtas todėl, nes buvo realizuota speciali *CSRF* apsauga, tinkanti veikimui su saityno programa, tad klaida buvo pažymėta kaip išspręsta.

Sekančioje klaidoje buvo nurodoma, jog yra naudojama potencialiai nesaugi *SHA256* maišos funkcija. Ši funkcija yra naudojama *PKCE* protokole sukuriant verifikavimo kodo santrauką. Pagal šio protokolo specifikaciją, *SHA256* yra tinkama ir pakankamai saugi funkcija, skirta sukurti reikiamų duomenų santrauką, tad aptikta saugumo klaida buvo pažymėta kaip išspręsta.

Likusios saugumo klaidos nurodė, jog reikia peržiūrėti saugumo anotacijas, esančias ant valdiklių metodų, kurios nurodo kokioms rolėms yra prieinami tam tikri prieigos taškai. Šioms anotacijoms ištestuoti buvo parašyti vienetų testai, tad aptiktos klaidos buvo pažymėtos kaip išspręstos.

4.1.3. Tarpinio įgaliotojo serverio programos kodo analizė

Analizuojant tarpinio įgaliotojo serverio programą, įrankis iš viso išanalizavo 388 kodo eilutes ir pateikė rezultatus, kurie yra pavaizduoti 72 pav.

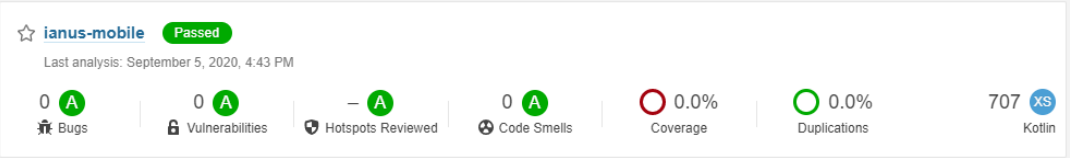


72 pav. Tarpinio įgaliotojo serverio programos statinės kodo analizės rezultatai

Vykdant analizę pirmą kartą, įrankis aptiko vieną saugumo klaidą, nurodydamas, jog yra naudojama potencialiai nesaugi *SHA256* maišos funkcija. Ši funkcija yra naudojama *PKCE* protokole sukuriant verifikavimo kodo santrauką. Pagal šio protokolo specifikaciją, *SHA256* yra tinkama ir pakankamai saugi funkcija, skirta sukurti reikiamų duomenų santrauką, tad aptikta saugumo klaida buvo pažymėta kaip išspręsta.

4.1.4. Mobiliojo ryšio įrenginio taikomosios programos kodo analizė

Analizuojant mobiliojo ryšio įrenginio taikomąją programą, įrankis iš viso išanalizavo 707 kodo eilutes ir pateikė rezultatus, kurie yra pavaizduoti 73 pav.



73 pav. Mobiliojo ryšio įrenginio taikomosios programos statinės kodo analizės rezultatai

Atlikus analizę, įrankis neaptiko saugumo klaidų ar netinkamo kodo vietų.

4.2. Autentifikavimo metodo kokybinis tyrimas

Siekiant nustatyti siūlomo autentifikavimo metodo kokybines charakteristikas, jis buvo tiriamas ir vertinamas pagal [65] literatūros šaltinyje siūlomą autentifikavimo schemų vertinimo metodiką.

Metodikoje yra pristatomos 25 autentifikavimo metodų charakteristikos, kuriomis yra įvertinami metodų privalumai ir trūkumai. Šaltinyje nurodoma, jog metodų saugumo ir panaudojamumo savybės būna sunkiai suderinamos – suteikiant vieną savybę, yra sudėtinga pritaikyti kitą. Autoriai pabrėžia ir dislokavimo savybę, nurodančią, ar metodas gali būti pritaikytas praktiškai. Šiame darbe yra siekiama pasiūlyti metodą, kuris būtų subalansuotas ir turėtų kuo geresnes saugumo, panaudojamumo ir pritaikymo savybes.

Atsižvelgus į išskirtas savybių grupes, charakteristikos buvo suskirstytos į tris kategorijas: panaudojamumas (angl. *usability*), dislokavimas (angl. *deployment*) ir saugumas (angl. *security*). Charakteristikų apibūdinimai ir jų realizavimo poveikis siūlomo metodo veikimui ir funkcionalumui yra pateikiami 2 lentelėje, 3 lentelėje ir 4 lentelėje.

2 lentelė. Panaudojamumo charakteristikų vertinimo analizės rezultatai

Nr.	**Charakteristika ir jos apibūdinimas**	**Charakteristikos įgyvendinimas siūlomame metode**
U1	Nereikalaujantis atminties pastangų (angl. *Memory-wise-Effortless*) Metodo naudotojai neturi prisiminti slaptų reikšmių (angl. *secrets*). Metodui skiriamas „±“ įvertinimas, jei naudotojas turi prisiminti tik vieną reikšmę.	Naudotojai turi prisiminti tik paskyros slaptažodį, kuomet yra sukonfigūruotas patvirtinimas naudojant piršto antspaudą. Jei yra sukonfigūruotas patvirtinimas naudojant *PIN* kodą, naudotojas taip pat turi prisiminti ir šią reikšmę. Kadangi siūlomą metodą rekomenduojama naudoti su sukonfigūruotu piršto antspaudo patvirtinimu, už šią charakteristiką metodui yra skiriamas „+“ vertinimas.
U2	Plečiamas tarp paskyrų (angl. *Scalable-for-Users*) Naudotojas gali taikyti metodą prisijungimui prie skirtingų sistemų paskyrų.	Metodas yra realizuotas vieningo prisijungimo sistemoje, tad naudotojas vieną paskyrą naudoja prisijungimui prie skirtingų sistemų. Už šią charakteristiką metodui yra skiriamas „+“ vertinimas.
U3	Nereikalaujantis papildomų nešulių (angl. *Nothing-to-Carry*) Naudojant metodą, naudotojas neturi papildomai nešiotis įvairių fizinių objektų. Metodui skiriamas „±“ įvertinimas, jei naudotojas turi nešiotis įrenginį, kurį ir taip visada nešiojasi (pvz. mobiliojo ryšio įrenginį).	Naudotojas su savimi turi turėti savo mobiliojo ryšio įrenginį, kurį dažniausiai visada nešiojasi su savimi. Už šią charakteristiką metodui yra skiriamas „±“ vertinimas.

Nr.	Charakteristika ir jos apibūdinimas	Charakteristikos įgyvendinimas siūlomame metode
U4	Nereikalaujantis fizinių pastangų (angl. *Physically-Effortless*) Metodo panaudojimas iš naudotojo nereikalauja didesnių fizinių pastangų nei mygtuko paspaudimas.	Naudotojas autentifikavimo proceso metu turi įvesti slaptažodį, paspausti kelis mygtukus ir pateikti *PIN* kodą ar piršto antspaudą, jei yra sukonfigūruotas užklausos patvirtinimas. Už šią charakteristiką metodui yra skiriamas „-„ vertinimas.
U5	Lengvai išmokstamas (angl. *Easy-to-Learn*) Naudotojai, neturėdami išankstinių žinių apie metodą, gali jį lengvai išmokti ir naudoti.	Vykdant autentifikacijos procesą, naudotojui yra pateikiamos išsamios instrukcijos ir paaiškinimai, kokius veiksmus jis turi atlikti, norint sėkmingai užbaigti visą procesą. Už šią charakteristiką metodui yra skiriamas „+“ vertinimas.
U6	Efektyviai naudojamas (angl. *Efficient-to-Use*) Laikas, kurį naudotojas praleidžia autentifikuodamasis, yra priimtinai trumpas. Laikas, kurį naudotojas skiria paskyros susiejimui, gali būti šiek tiek ilgesnis.	Autentifikacijos procesas gali užtrukti, kadangi vadovas turi sureaguoti ir autorizuoti užklausą. Mobiliojo ryšio įrenginio susiejimo procesas yra pakankamai greitas. Atsižvelgiant į tai, kad autentifikacijos procesui užbaigti reikalingi patvirtinimai iš kelių asmenų, už šią charakteristiką metodui yra skiriamas „-“ vertinimas.
U7	Retai klaidingas (angl. *Infrequent-Errors*) Veiksmai, kuriuos naudotojas turi atlikti vykdydamas metodą, dažniausiai atliekami sėkmingai, kuomet juos atlieka teisėtas naudotojas. Schema nėra sudėtinga naudoti ir yra patikima.	Autentifikavimo schema nėra sudėtinga ir turėtų visada suveikti, kuomet ją vykdo teisėtas naudotojas. Kuomet užklausa yra patvirtinama panaudojant biometrinius piršto antspaudo duomenis, yra naudojami gamykliniai mobiliųjų ryšio įrenginių mechanizmai, tad schemos patikimumas priklauso ir nuo šių technologijų patikimumo. Už šią charakteristiką metodui yra skiriamas „±“ vertinimas.
U8	Lengvai kompensuojamas įvykus nelaimei (angl. *Easy-Recovery-from-Loss*) Naudotojas gali patogiai atgauti galimybę autentifikuotis, kuomet įvyksta nelaimė - pamirštamas slaptažodis ar pametamas žetonas.	Sukūrus paskyrą, naudotojui yra perduodami keli vienkartiniai slaptažodžiai, su kuriais jis nelaimės atveju gali prisijungti prie paskyrų valdymo sistemos. Prisijungus prie šios sistemos, galima pasikeisti slaptažodį ar pakeisti įrenginį, su kuriuo paskyra yra susieta. Už šią charakteristiką metodui yra skiriamas „+“ vertinimas.

3 lentelė. Dislokavimo charakteristikų vertinimo lentelė

Nr.	Charakteristika ir jos apibūdinimas	Charakteristikos įgyvendinimas siūlomame metode
D1	Prieinamumas (angl. *Accessibility*) Naudotojai, turintys tam tikrą fizinę negalią, gali sėkmingai naudotis metodu.	Autentifikacijos schema remiasi *WEB* naršyklės ir mobiliojo ryšio įrenginio teikiamomis standartinėmis prieinamumo priemonėmis. Papildomų veiksmų, kuriuos naudotojui potencialiai būtų sudėtinga atlikti, nėra. Už šią charakteristiką metodui yra skiriamas „+“ vertinimas.
D2	Nežymi kaina už kiekvieną naudotoją (angl. *Negligible-Cost-per-User*) Kaina, kurią reikia sumokėti už kiekvienos paskyros galimybę taikyti metodą, yra nežymi. Į kainą įeina įrenginiai ar prietaisai, kuriuos turi išduoti metodo tiekėjas.	Metodui realizuoti nėra reikalinga papildoma techninė įranga – yra naudojami naudotojų mobiliojo ryšio įrenginiai. Už šią charakteristiką metodui yra skiriamas „+“ vertinimas.

Nr.	Charakteristika ir jos apibūdinimas	Charakteristikos įgyvendinimas siūlomame metode
D3	Suderinamas su serveriu (angl. *Server-Compatible*) Tiekėjui, norinčiam teikti metodą, nereikia keisti egzistuojančios programinės įrangos ar infrastruktūros. Metodui skiriamas „±“ vertinimas, jei jis yra realizuojamas *SSO* tipo sistemoje, kuri gali integruotis su kitomis sistemomis.	Metodas yra realizuojamas naudojant specialiai sukurtą programinę įrangą, kuri gali būti diegiama standartiniuose debesijos serveriuose. Metodas yra realizuotas vieningo prisijungimo sistemoje, taikančioje standartizuotą *OAuth* 2.0 protokolą, kuris yra suderinamas su įvairiais paslaugų tiekėjais. Atsižvelgus, jog ne visi paslaugų tiekėjai naudoja standartizuotą *OAuth* 2.0 protokolą, už šią charakteristiką metodui yra skiriamas „±“ vertinimas.
D4	Suderinamas su naršykle (angl. *Browser-Compatible*) Naudotojai, norėdami panaudoti metodą, neturi keisti savo dabartinių programinės įrangos klientų. Metodas turėtų veikti, kuomet yra naudojama naujausia, standartus atitinkanti naršyklė, palaikanti *HTML5* ir *JavaScript* technologijas. Naudotojai taip pat nėra įpareigoti diegti papildomus plėtinius ar papildomą programinę įrangą (pvz. taikomąsias mobiliojo ryšio įrenginio programas).	Sistemos prototipui, taikančiam šį metodą, buvo iškeltas nefunkcinis reikalavimas, jog saityno programa turi teisingai veikti naudojant *Google Chrome* naršyklę su N-2 versijomis. Naudotojai taip pat papildomai turi įsidiegti mobiliojo ryšio įrenginio taikomąją programą. Kadangi naudotojai į mobiliojo ryšio įrenginį turi įsidiegti papildomą taikomąją programą, už šią charakteristiką metodui yra skiriamas „±“ vertinimas.
D5	Brandus (angl. *Mature*) Metodas buvo įgyvendintas ir įdiegtas plataus masto pritaikymui.	Atsižvelgus, jog šis metodas yra dar tik siūlomas, už šią charakteristiką metodui yra skiriamas „-“ vertinimas.
D6	Neapsaugotas patento (angl. *Non-Proprietary*) Metodas gali būti laisvai taikomas nemokant honorarų ar kitų mokesčių.	Atsižvelgus, jog šis metodas yra pateikiamas moksliniame darbe ir nėra patentuotas, už šią charakteristiką metodui yra skiriamas „+“ vertinimas.

4 lentelė. Saugumo charakteristikų vertinimo analizės rezultatai

Nr.	Charakteristika ir jos apibūdinimas	Charakteristikos įgyvendinimas siūlomame metode
S1	Atsparus fiziniam stebėjimui (angl. *Resilient-to-Physical-Observation*) Piktavalis negali apsimesti naudotoju stebėdamas kaip jis atlieka autentifikavimo procesą. Atakos apima stebėjimą per petį ar klaviatūros paspaudimų fiksavimą.	Metodas yra atsparus fiziniam stebėjimui, kadangi norint atlikti autentifikacijos procesą, reikia turėti fizinį mobiliojo ryšio įrenginį ir pateikti piršto antspaudą, kuomet yra sukonfigūruotas užklausos patvirtinimas pateikiant šią biometrinę informaciją. Už šią charakteristiką metodui yra skiriamas „+“ vertinimas.
S2	Atsparus nutaikytam apsimetinėjimui (angl. *Resilient-to-Targeted-Impersonation*) Piktavalis negali apsimesti naudotoju, pasinaudodamas žiniomis apie taikinį - jo gimimo data, asmenine informacija, informacija apie artimuosius ar kita privačia informacija.	Metodas yra atsparus nutaikytam apsimetinėjimui, kadangi nėra naudojama asmeninė naudotojo informacija. Už šią charakteristiką metodui yra skiriamas „+“ vertinimas.
S3	Atsparus apribotam spėliojimui (angl. *Resilient-to-Throttled-Guessing*) Piktavalis, kurio paslapčių spėliojimo greitis yra apribotas metodą realizuojančios sistemos, negali atspėti paslapties per tam tikrą prasmingą laiką.	Metodas yra atsparus apribotam spėliojimui, kadangi metodą realizuojančiam prototipui yra keliamas nefunkcinis reikalavimas: „1.1. Nesėkmingai atlikus 3 bandymus prisijungti prie paskyros, paskyra yra užblokuojama. Užblokuotą paskyrą atblokuoti gali tik administratorius“ Už šią charakteristiką metodui yra skiriamas „+“ vertinimas.

Nr.	Charakteristika ir jos apibūdinimas	Charakteristikos įgyvendinimas siūlomame metode
S4	Atsparus neapribotam spėliojimui (angl. *Resilient-to-Unthrottled-Guessing*) Piktavalis, kurio paslapčių spėliojimo greitis yra apribotas tik naudojamos techninės/programinės įrangos greitaveika, negali atspėti paslapties per tam tikrą prasmingą laiką.	Metodas yra atsparus neapribotam spėliojimui, kadangi metodą realizuojančiam prototipui yra keliamas nefunkcinis reikalavimas: „1.1. Nesėkmingai atlikus 3 bandymus prisijungti prie paskyros, paskyra yra užblokuojama. Užblokuotą paskyrą atblokuoti gali tik administratorius“ Už šią charakteristiką metodui yra skiriamas „+“ vertinimas.
S5	Atsparus vidiniam stebėjimui (angl. *Resilient-to-Internal-Observation*) Piktavalis negali apsimesti naudotoju perimdamas įvestį iš naudojamų įrenginių (pvz. naudojant klaviatūros paspaudimus fiksuojančią programinę įrangą) arba perimant tinklo srautą (pvz. pažeidus *TLS* sujungimą). Metodas turi būti apsaugotas nuo pakartojimo atakų. Dviejų faktorių metodams, kuriems žalinga programine įranga paveikus du faktorius metodas tampa pažeidžiamas, suteikiamas „±“ vertinimas.	Metodas yra atsparus pakartojimo atakoms, kadangi vykdant autentifikacijos procesą yra tikrinama, kad proceso žingsniai būtų atliekami tik vieną kartą. Metodas gali būti pažeistas, jei žalinga programine įranga bus paveikta naudotojo naršyklė, naudotojo ir vadovo mobiliojo ryšio įrenginiai. Kadangi metodo kompromitacijai reikėtų žalinga programine įranga paveikti net tris įrenginius, už šią charakteristiką metodui yra skiriamas „+“ vertinimas.
S6	Atsparus informacijos nutekėjimui iš tikrintojo (angl. *Resilient-to-Leaks-from-Other-Verifiers*) Metodas užtikrina, jog informacijai nutekėjus iš tikrintojo, piktavaliai negalės pasinaudoti šia informacija apsimetant naudotoju.	Vieningo prisijungimo sistema (tikrintojas) naudotojų slaptažodžius saugo naudodama stiprų *bcrypt* slaptažodžių santraukos algoritmą. Sukompromitavus maišos reikšmę, slaptažodis galėtų būti atkurtas tik taikant grubios jėgos perrinkimo metodą. Vieningo prisijungimo sistema taip pat saugo tik viešąjį naudotojo kriptografinį raktą, kuris yra skirtas patvirtinti užklausų patvirtinimo parašus, tad šio rakto kompromitacija nepaveiktų metodo saugos. Privatusis naudotojo kriptografinis raktas yra saugomas tik naudotojo mobiliojo ryšio įrenginyje. Už šią charakteristiką metodui yra skiriamas „+“ vertinimas.
S7	Atsparus apgaulės atakoms (angl. *Resilient-to-Phishing*) Piktavalis negali perimti naudotojo paslapčių naudodamas netikrus tikrintojo tinklalapius (pvz. pažeisdamas *DNS* paslaugą).	Piktavalis apgaulės metodu galėtų perimti tik naudotojo slaptažodį, tačiau negalėtų paveikti užklausų patvirtinimo naudojant mobiliojo ryšio įrenginį. Kadangi apgaulės atakos metodu gali būti pažeista tik viena autentifikacijos metodo sudedamoji dalis, už šią charakteristiką metodui yra skiriamas „+“ vertinimas.
S8	Atsparus vagystei (angl. *Resilient-to-Theft*) Kuomet metodas naudoja fizinį įrenginį autentifikacijos atlikimui, piktavalio perimtas įrenginys negali būti panaudotas autentifikacijos vykdymui.	Kuomet užklausų patvirtinimui yra sukonfigūruotas *PIN* kodo panaudojimas, jį perrinkti grubios atakos metodu yra lengviau. Tačiau, jei yra sukonfigūruotas piršto antspaudo panaudojimas, įrenginio panaudojimas tampa labai komplikuotas. Taip pat svarbu paminėti, jog perėmus ir panaudojus tik naudotojo įrenginį, metodui apsaugą suteikia vadovo įrenginio panaudojimas autorizacijos patvirtinimui. Už šią charakteristiką metodui yra skiriamas „+“ vertinimas.

Nr.	Charakteristika ir jos apibūdinimas	Charakteristikos įgyvendinimas siūlomame metode
S9	Nenaudojantis patikimos trečios šalies (angl. *No-Trusted-Third-Party*) Metodas nesiremia trečios šalies patikimumu, kurios saugos pažeidimas sukompromituotų metodo saugą.	Metodą realizuojanti vieningo prisijungimo sistema remiasi trečiosios šalies *Google Firebase* paslauga. Informacija, kuri yra siunčiama šiai paslaugai, yra užšifruojama naudotojo viešuoju kriptografiniu raktu, tad yra užtikrinamas konfidencialumas. Visgi, trečiosios šalies prieinamumo ar integralumo trikdžiai gali paveikti autentifikacijos metodo saugą, tad už šią charakteristiką metodui yra skiriamas „-“ vertinimas.
S10	Reikalaujantis tiesioginio patvirtinimo (angl. *Requiring-Explicit-Consent*) Autentifikacijos procesas negali būti pradėtas prieš tai naudotojui nepateikus tiesioginio prašymo.	Naudotojas, norėdamas pradėti autentifikacijos procesą, turi tiesiogiai pareikšti prašymą, pateikiant pradinę prisijungimo vardo ir slaptažodžio formą. Už šią charakteristiką metodui yra skiriamas „+“ vertinimas.
S11	Neatsiejamas (angl. *Unlinkable*) Tai yra privatumo paslauga, nurodanti, ar paslaugos tiekėjai gali atsekti, jog tas pats naudotojas autentifikuojasi pas skirtingus paslaugos tiekėjus. Išimtis yra naudotojo identifikatoriaus ar *IP* adreso pateikimas.	*JWT* žetone, kurį gauna paslaugos tiekėjai, yra nurodomas vieningo prisijungimo sistemos naudotojui suteiktas unikalus identifikatorius. Kadangi charakteristikos apraše naudotojo identifikatorius yra pateikiamas kaip išimtis, už šią charakteristiką metodui yra skiriamas „+“ vertinimas.

5 lentelėje yra pateikiamas kitų mokslininkų darbuose siūlomų autentifikavimo metodų sąrašas, kurių kokybinės charakteristikos taip pat bus įvertintos panaudojant [65] šaltinyje nurodytą metodiką.

5 lentelė. Lyginamų autentifikavimo metodų sąrašas

Metodo Nr.	Metodo apibūdinimas	Literatūros šaltiniai
1.	Piršto antspaudo panaudojimu ir naudotojo vietove paremtas kelių faktorių autentifikavimo metodas skirtas mobiliojo ryšio įrenginiams	[28]
2.	*LocBiometrics*: Mobiliojo ryšio įrenginiu grindžiama kelių faktorių biometrinė autentifikacija su laiko ir vietos užtikrinimu	[66]
3.	Naujas stiprus slaptažodžio generatorius skirtas debesijos autentifikavimo gerinimui	[48]
4.	Internetinės bankininkystės prisijungimas naudojant kelių faktorių autentifikavimą	[67]
5.	Kelių faktorių autentifikavimo protokolas mobiliojo ryšio įrenginių aplinkoje	[68]
6.	Internetinės bankininkystės autentifikavimas naudojant mobiliojo ryšio įrenginius	[69]
7.	Unikaliu identifikatoriumi grįsta kelių faktorių autentifikavimo schema skirta e-paslaugoms	[70]
8.	Kelių faktorių mechanizmo, skirto saugiai autentifikavimo sistemai, projektavimas ir įgyvendinimas	[71]
9.	Siūlomas metodas	

Aukščiau išvardintų autentifikavimo metodų apibendrinti kokybinės analizės rezultatai pateikiami 6 lentelėje. Siekiant metodų kokybinius įverčius palyginti tarpusavyje, buvo pritaikytas skaitinis charakteristikų įgyvendinimo įvertinimo metodas:

– Už „+“ įvertinimą skiriamas vienas balas;
– Už „-“ įvertinimą skiriama nulis balų;
– Už „±“ įvertinimą skiriama pusė balo.

Atsižvelgus, jog metodą siekiama taikyti aukšto saugos lygio reikalaujančioje kritinės infrastruktūros aplinkoje, metodo saugumo charakteristikos yra svarbesnės nei panaudojamumas ar dislokavimas, tad saugumo charakteristikų balų sumai yra suteikiamas didesnis koeficientas. Iš viso metodas daugiausiai gali surinkti 30,5 balo. Paskutiniame 6 lentelės stulpelyje yra pateikiama bendra balų suma, apskaičiuojama pagal 1 formulėę.

$$Bendras\ balas = \sum_{i}^{8} U_i + \sum_{j}^{6} D_j + 1{,}5 \cdot \sum_{j}^{11} S_k \qquad (1)$$

Čia:
$U_i - i - tosios\ panaudojamumo\ charakteristikos\ balas,$
$D_j - j - tosios\ dislokavimo\ charakteristikos\ balas,$
$S_k - k - tosios\ saugumo\ charakteristikos\ balas.$

1 formulė. Metodo kokybinės charakteristikos skaitinio įverčio formulė

6 lentelė. Autentifikavimo metodų kokybinės analizės rezultatai

Nr.	Charakteristikų įgyvendinimas																								Įvertis
	Panaudojamumas (*U*)								Dislokavimas (*D*)						Saugumas (*S*)										
	1	2	3	4	5	6	7	8	1	2	3	4	5	6	1	2	3	4	5	6	7	8	9	10	11	
1.	1	1	1	0	1	1	.5	0	1	1	0	.5	0	1	1	.5	1	1	0	1	1	1	1	1	1	23.25
2.	0	0	1	0	1	0	0	0	1	1	0	.5	0	1	1	.5	1	1	.5	1	1	1	0	1	1	19
3.	0	1	.5	0	1	.5	1	0	0	1	0	.5	0	1	1	1	1	1	.5	0	1	0	1	1	1	19.25
4.	0	1	1	0	1	1	1	0	1	1	0	1	1	1	0	1	1	0	0	0	0	1	1	1	1	19
5.	1	1	.5	0	1	.5	.5	0	1	1	0	.5	1	1	1	1	1	1	.5	1	1	1	0	1	1	23.25
6.	1	1	.5	0	1	1	1	0	1	1	0	.5	0	1	1	1	1	1	0	1	1	.5	1	1	1	23.25
7.	0	1	.5	0	1	.5	1	0	0	1	.5	.5	0	1	0	1	1	1	.5	1	1	1	0	1	1	19.75
8.	0	1	.5	0	1	.5	1	0	1	1	0	1	1	1	1	1	1	1	.5	0	0	1	0	1	1	20.25
9.	**1**	**1**	**.5**	**0**	**1**	**0**	**.5**	**1**	**1**	**1**	**.5**	**.5**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**24**

Apibendrinant rezultatus, pateiktus 6 lentelėje, galima teigti, jog keli kitų autorių siūlomi autentifikavimo metodai surinko nemažą kiekį balų – pirmasis, penktasis ir šeštasis metodai surinko 23,25 balus. Taikant skaitinį charakteristikų įgyvendinimo įvertinimo metodą, siūlomas autentifikavimo metodas buvo įvertintas aukščiausiai (24 balai), lyginant su kitais tirtais metodais. Iš to galima daryti išvadą, kad siūlomas autentifikavimo metodas yra kokybiškesnis vertintų kategorijų visumos atžvilgiu.

Siūlomas metodas gavo 0 balų už *U6* charakteristiką (Efektyviai naudojamas), tačiau tarp kitų metodų išsiskyrė tuo, jog už *S5* charakteristiką (Atsparus vidiniam stebėjimui) gavo vieną balą, kadangi pažeisti visus 3 įrenginius, naudojamus siūlomame autentifikacijos procese, yra sunku ir reikalauja daug pastangų.

Siūlomas metodas už panaudojamumo charakteristikas iš viso surinko 5 balus, nedaug atsilikdamas nuo didžiausią panaudojamumo balą (5,5) gavusių metodų. Už dislokavimo charakteristikas metodas iš viso surinko 4 balus, tuo tarpu didžiausias dislokavimo balas buvo 5.

Galiausiai, už saugumo charakteristikas metodas surinko didžiausią balą iš visų nagrinėtų metodų – 15, kuomet antras pagal dydį balas buvo 14,25.

4.3. *WEB* serverių konfigūracijos analizė

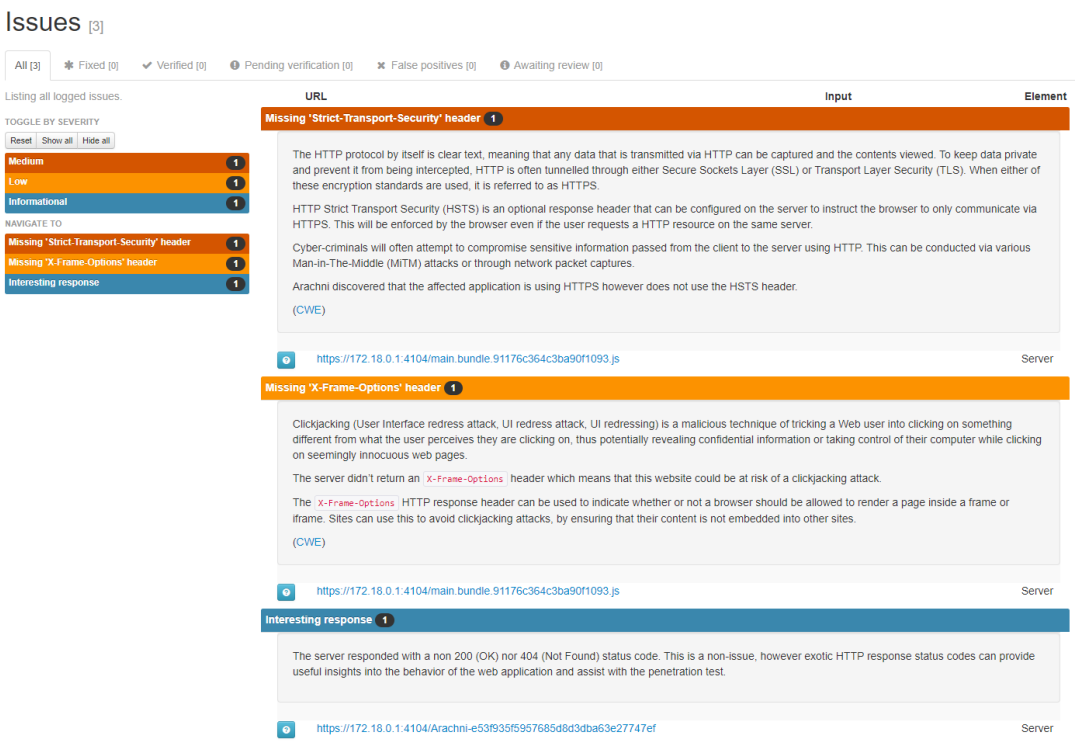
Siekiant užtikrinti sistemos saugą, yra svarbu garantuoti, jog serverių konfigūracija yra saugi ir atitinka gerąsias praktikas. Prototipe naudojamų serverių konfigūracijos saugos analizei atlikti buvo pasitelkti dinaminės ir statinės analizės metodai, kurių rezultatai yra pateikiami šiame poskyryje.

4.3.1. Dinaminė analizė

Siekiant patikrinti, ar *UI/Gateway* ir *Proxy WEB* serveriai yra sukonfigūruoti teisingai ir saugiai, buvo atliekama dinaminė *WEB* serverių saugumo analizė. Analizė buvo atliekama pasitelkus šiuos įrankius:

- *Arachni* 1.5.1[1]
- *Nikto* v2.1.6[2]
- *OWASP ZAP* 2.9.0[3]
- *Subgraph Vega* 1.0[4]

74 pav. yra pateikta pavyzdinė rezultatų, kuriuos pateikė *Arachni* įrankis, ekrano kopija. Kitų įrankių pateiktų rezultatų ekrano kopijos yra pateikiamos 4 priede.



74 pav. *UI/Gateway* serverio konfigūracijos analizės rezultatai panaudojant *Arachni* įrankį

[1] https://www.arachni-scanner.com/
[2] https://github.com/sullo/nikto
[3] https://www.zaproxy.org/
[4] https://subgraph.com/vega/index.en.html

Panaudojus aukščiau nurodytus įrankius dinaminės analizės atlikimui, gauti rezultatai buvo susisteminti ir pateikti 7 lentelėje, 8 lentelėje, ir 9 lentelėje. Šiose lentelėse taip pat yra pateikiamos pažeidžiamumų būsenos ir pašalinimo sprendimai.

7 lentelė. *UI/Gateway* serverio konfigūracijos analizės rezultatai

Problema Nr. 1	
Problemos pavadinimas	Trūkstama *Strict-Transport-Security* antraštė
Aptikta įrankio	*Arachni*
Problemos apibūdinimas	Serverio grąžinama neprivaloma *HTTP Strict-Transport-Security* antraštė naršyklei nurodo, jog naršyklė su serveriu gali komunikuoti tik naudojant saugų *HTTPS* protokolą. Gavusi tokį paliepimą iš serverio, naršyklė užtikrins, jog į nurodytą serverį nebus leidžiama atlikti *HTTP* užklausų. *Arachni* aptiko, jog serveris naudoja *HTTPS* protokolą, tačiau nenaudoja *HSTS* antraštės.
Susiję URL	*/main.bundle.91176c364c3ba90f1093.js*
Pavojingumas (angl. severity)	Vidutinis
Būsena	✔ Išspręsta
Komentaras	Trūkstama antraštė buvo pridėta į *nginx.conf* konfigūracijos failą.
Problema Nr. 2	
Problemos pavadinimas	Trūkstama *X-Frame-Options* antraštė
Aptikta įrankio	*Arachni*
Problemos apibūdinimas	Serveris negrąžino *HTTP X-Frame-Options* antraštės, ko pasekoje tinklalapis gali būti pažeistas naudojant *clickjacking* ataką. Ši antraštė naršyklei nurodo, ar tinklalapis gali būti atvaizduojamas rėme (angl. *iframe*). Serveriai naudoja šią antraštę siekiant apsisaugoti nuo atakų, kuomet tinklalapiai yra įterpiami į kitų pavojingų tinklalapių rėmus.
Susiję URL	*/main.bundle.91176c364c3ba90f1093.js*
Pavojingumas (angl. severity)	Žemas
Būsena	✔ Išspręsta
Komentaras	Trūkstama antraštė buvo pridėta į *nginx.conf* konfigūracijos failą.
Problema Nr. 3	
Problemos pavadinimas	Įdomus atsakymas
Aptikta įrankio	*Arachni*
Problemos apibūdinimas	Serveris į *URL* atsakė su nestandartiniais statuso kodais.
Susiję URL	*/Arachni-e53f935f5957685d8d3dba63e27747ef*
Pavojingumas (angl. severity)	Informacinis
Būsena	✔ Klaidingai teigiama
Komentaras	Serveris yra sukonfigūruotas teikti vieno puslapio programą (angl. *single-page application*), tad tokia serverio elgsena yra teisinga.
Problema Nr. 4	
Aptikta įrankio	*Nikto*

Problemos apibūdinimas	Serveris naudoja *TLS* protokolą, tačiau nėra pateikta *HTTP Expect-CT* antraštė.
Būsena	✔ Išspręsta
Komentaras	Trūkstama antraštė buvo pridėta į *nginx.conf* konfigūracijos failą.
Problema Nr. 5	
Problemos pavadinimas	Kliento šifravimo rinkinių pirmumas
Aptikta įrankio	*Vega*
Problemos apibūdinimas	Serveris gali nepaisyti kliento šifravimo rinkinių pirmumo per *TLS* rankų paspaudimo fazę. Tai yra naudinga, kadangi serveris gali liepti klientams naudoti geresnius ir saugesnius šifravimo rinkinius. *Vega* aptiko, jog šis funkcionalumas serveryje nėra sukonfigūruotas, ko pasekoje naudotojų naršyklės gali pasirinkti mažiau saugius šifravimo rinkinius ir atverti galimybes įvairioms atakoms.
Pavojingumas (angl. severity)	Vidutinis
Būsena	✔ Išspręsta
Komentaras	Serveris buvo sukonfigūruotas pateikti siūlomų saugių šifravimo algoritmų sąrašą, algoritmus nurodant *nginx.conf* konfigūracijos faile esančioje *ssl_ciphers* direktyvoje.
Problema Nr. 6	
Problemos pavadinimas	Nėra palaikomas tolimesnis (angl. *forward*) slaptumas
Aptikta įrankio	*Vega*
Problemos apibūdinimas	*Vega* aptiko, jog serveris nepalaiko ar neteikia pirmenybės tolimesnio slaptumo šifrų rinkiniams. Tolimesnio slaptumo šifrai naudoja tokius algoritmus kaip efemerinį (trumpalaikį) *Diffie-Hellman* algoritmą, kuris sukuria vienkartinį sesijos raktą. Šis raktas nėra išvestas iš ilgalaikio privataus rakto, tad užšifruotų duomenų saugumas nepriklauso nuo to, ar raktas bus sukompromituotas ateityje. Jei nėra naudojamas tolimesnio slaptumo šifras, sesijos duomenys yra saugūs tol, kol yra saugus serverio ilgalaikis privatus raktas.
Pavojingumas (angl. severity)	Vidutinis
Būsena	✔ Išspręsta
Komentaras	Serveris buvo sukonfigūruotas pateikti norimą saugių šifravimo algoritmų sąrašą, algoritmus nurodant *nginx.conf* konfigūracijos faile esančioje *ssl_ciphers* direktyvoje. Tarp algoritmų yra nurodomi saugūs efemeriniai *Diffie-Hellman* šeimos algoritmai.

8 lentelė. *Proxy* serverio konfigūracijos analizės rezultatai

Problema Nr. 7	
Problemos pavadinimas	Bendra direktorija
Aptikta įrankio	*Arachni*
Problemos apibūdinimas	Įrankis aptiko kelis dažnai naudojamų direktorijų vardus, kurie potencialiai gali būti nebenaudojami, tačiau vis dar yra viešai pasiekiami.
Susiję URL	*/home* */config*

Pavojingumas (angl. severity)	Vidutinis
Būsena	✔ Klaidingai teigiama
Komentaras	Aptiktos direktorijos yra teisingos ir naudojamos.
Problema Nr. 8	
Problemos pavadinimas	Bendri jautrūs failai
Aptikta įrankio	*Arachni*
Problemos apibūdinimas	Įrankis aptiko kelis dažnai naudojamų failų vardus, kurie potencialiai gali būti nebenaudojami, tačiau vis dar yra viešai pasiekiami.
Susiję URL	*/robots.txt*
Pavojingumas (angl. severity)	Žemas
Būsena	✔ Klaidingai teigiama
Komentaras	Aptikti failai yra teisingi ir naudojami.
Problema Nr. 9	
Problemos pavadinimas	Įdomus atsakymas
Aptikta įrankio	*Arachni*
Problemos apibūdinimas	Serveris į *URL* atsakė su nestandartiniais statuso kodais.
Susiję URL	*/authorize/?%3Cmy_tag_f2edfb 249980839587b136b1bf2543f5/%3E=* */authorize* */* */Arachni-f2edfb249980839587b136b1bf2543f5* */assets/images/*
Pavojingumas (angl. severity)	Informacinis
Būsena	✔ Klaidingai teigiama
Komentaras	Užklausų atsakymai buvo patikrinti ir buvo nustatyta, jog tai yra teisinga serverio elgsena.
Problema Nr. 10	
Aptikta įrankio	*Nikto*
Problemos apibūdinimas	*HTTP X-XSS-Protection* antraštės reikšmė nurodo, jog yra išjungta apsauga nuo *XSS* tipo atakų.
Būsena	✔ Klaidingai teigiama
Komentaras	*X-XSS-Protection* antraštė nebėra naudojama naujų naršyklių, kurios palaiko *CSP* apsaugos mechanizmą.
Problema Nr. 11	
Aptikta įrankio	*Nikto*
Problemos apibūdinimas	Aptikta nestandartinė *HTTP x-dns-prefetch-control* antraštė su reikšme *off*.
Būsena	✔ Klaidingai teigiama
Komentaras	Tai yra teisinga *HTTP* antraštė[5], kurią prideda *Helmet*[6] biblioteka.
Problema Nr. 12	

[5] https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-DNS-Prefetch-Control

[6] https://helmetjs.github.io/

Aptikta įrankio	*Nikto*
Problemos apibūdinimas	*HTTP Expect-CT* antraštės reikšmė nepriverčia nutraukti sujungimo, kuomet sertifikato skaidrumo įrašai (angl. *Certificate Transparency Log*) yra neteisingi.
Būsena	✔ Išspręsta
Komentaras	*HTTP Expect-CT* antraštės reikšmė buvo pataisyta.
Problema Nr. 13	
Aptikta įrankio	*Nikto*
Problemos apibūdinimas	Aptikta *HTTP x-powered-by* antraštė su reikšme *Express*.
Būsena	✔ Išspręsta
Komentaras	*HTTP* antraštė buvo pašalinta.
Problema Nr. 14	
Aptikta įrankio	*Nikto*
Problemos apibūdinimas	Aptiktas */config/ URL*, kuris potencialiai suteikia nuotolinį valdymą.
Būsena	✔ Klaidingai teigiama
Komentaras	*URL* buvo patikrintas ir buvo nustatyta, jog *URL* teikiamas funkcionalumas yra teisingas - naudojama *GET* užklausa gauti bazinę informaciją apie klasterio būseną.
Problema Nr. 15	
Aptikta įrankio	*Nikto*
Problemos apibūdinimas	Aptiktas įdomus */home/ URL*.
Būsena	✔ Klaidingai teigiama
Komentaras	*URL* buvo patikrintas ir buvo nustatyta, jog *URL* teikiamas funkcionalumas yra teisingas – naudojama *GET* užklausa gauti pagrindinio puslapio *HTML* kodą.
Problema Nr. 16	
Problemos pavadinimas	*CSP*: Pranešimai
Aptikta įrankio	*ZAP*
Problemos apibūdinimas	Įspėjimai: 1:36: *block-all-mixed-content* direktyva yra eksperimentinė direktyva, kuri tikėtina, jog bus pridėta į *CSP* specifikaciją. 1:232: *upgrade-insecure-requests* direktyva yra eksperimentinė direktyva, kuri tikėtina, jog bus pridėta į *CSP* specifikaciją.
Pavojingumas (angl. severity)	Žemas
Būsena	✔ Klaidingai teigiama
Komentaras	Sistemai yra keliamas nefunkcinis reikalavimas „1.6 Saityno programa turi teisingai veikti naudojant *Google Chrome* naršyklę su N-2 versijomis“. Šiuo metu naujausia *Chrome* 88.0.4324.146 versija palaiko šią direktyvą, tad problema nėra aktuali.
Problema Nr. 17	
Problemos pavadinimas	Aptiktas lokalus failų sistemos kelias
Aptikta įrankio	*Vega*

Problemos apibūdinimas	*Vega* aptiko galimą absoliutų failų sistemos kelią. Ši informacija yra jautri, kadangi gali atskleisti informaciją apie serverio veikimo aplinką. Žinios apie failų sistemos išdėstymą gali padidinti galimybes atlikti aklas atakas.
Susijusios užklausos	*GET /scripts.391d299173602e261418.js*
Pavojingumas (angl. severity)	Vidutinis
Būsena	✔ Klaidingai teigiama
Komentaras	Šis failas yra generuojamas *Webpack* pakavimo įrankio. Failo turinys buvo peržiūrėtas ir buvo nustatyta, jog kelias neatskleidžia jautrios serverio konfigūracijos informacijos.

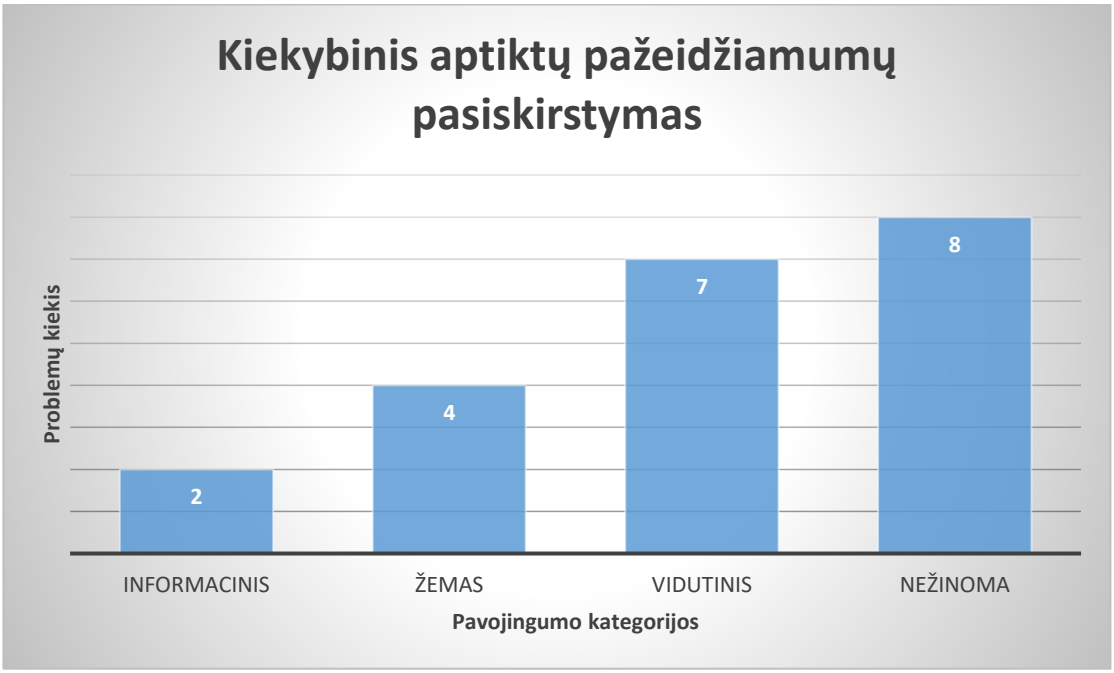
Dalis įrankių aptiko tokias pat problemas tiek *UI/Gateway*, tiek *Proxy* serveriuose. Siekiant išvengti dubliavimo, šie rezultatai yra pateikiami 9 lentelė**.**

9 lentelė. Bendri *UI/Gateway* ir *Proxy* serverių konfigūracijos analizės rezultatai

Problema Nr. 18	
Problemos pavadinimas	*CSP*: Viską leidžianti direktyva
Aptikta įrankio	*ZAP*
Problemos apibūdinimas	Žemiau nurodytos direktyvos leidžia visus turinio šaltinius: *frame-ancestors, form-action* Šios direktyvos nepaveldi reikšmių iš *default-src* direktyvos, tad nenurodžius jų reikšmių yra leidžiami visi turinio šaltiniai.
Pavojingumas (angl. severity)	Vidutinis
Būsena	✔ Išspręsta
Komentaras	Trūkstamos *CSP* direktyvos buvo pridėtos.
Problema Nr. 19	
Aptikta įrankio	*Nikto*
Problemos apibūdinimas	*HTTP Content-Encoding* antraštė turi *deflate* reikšmę. Tai reiškia, jog serveris yra jautrus *BREACH* tipo atakai.
Būsena	✔ Klaidingai teigiama
Komentaras	Užklausos buvo peržiūrėtos ir buvo nustatyta, jog visi *HTTP* atsakymai turi antraštę *Content-Encoding: gzip*.
Problema Nr. 20	
Problemos pavadinimas	*CSP*: *style-src unsafe-inline* direktyva
Aptikta įrankio	*ZAP*
Problemos apibūdinimas	*CSP style-src* direktyva įtraukia nesaugią *unsafe-inline* reikšmę.
Pavojingumas (angl. severity)	Vidutinis
Būsena	✕ Nebus taisoma
Komentaras	*UI* programa naudoja *styled-components* biblioteką, kuri *CSS* stilius įterpia tiesiogiai į *<script> HTML* blokus, tad norint, jog naršyklė sėkmingai užkrautų šiuos stilius, reikalinga *unsafe-inline* reikšmė. Ši problema galėtų būti ištaisyta pritaikant generuojamų *nonce* mechanizmą, tačiau tai gali būti taikoma tik tose programose, kurių puslapiai yra generuojami serverio pusėje. Kadangi ši programa yra generuojama kliento pusėje, tokio problemos sprendimo pritaikyti negalima.

Problema Nr. 21	
Problemos pavadinimas	Nepilnos ar trūkstamos *Cache-control* ir *Pragma HTTP* antraštės
Aptikta įrankio	*ZAP*
Problemos apibūdinimas	*Cache-control* ir *Pragma HTTP* antraščių reikšmės nebuvo nurodytos teisingai arba jų nėra iš viso, ko pasekoje naršyklė ir tarpiniai serveriai turinį kelia į podėlį (angl. *cache*).
Pavojingumas (angl. severity)	Žemas
Būsena	✔ Išspręsta
Komentaras	Trūkstamos *HTTP* antraštės buvo pridėtos.

Grafike, kuris pateikiamas 75 pav., yra nurodomas apibendrintas aptiktų pažeidžiamumų pasiskirstymas pagal pavojingumo lygį. Į „Nežinoma“ kategoriją buvo įtraukti *Nikto* įrankio aptikti pažeidžiamumai, kadangi įrankis prie aptiktų problemų nenurodė jų pavojingumo kategorijos.



75 pav. Aptiktų problemų pavojingumo kategorijų pasiskirstymas

Grafike, kuris pateikiamas 76 pav., yra nurodoma informacija apie išanalizuotų pažeidžiamumų būsenas. Matoma, jog didžioji dalis aptiktų problemų buvo klaidingai teigiamos arba ištaisytos. Viena aptikta problema nebus taisoma, kadangi prototipo realizacijai pasirinktos technologijos apriboja šios problemos išsprendimą.

76 pav. Išanalizuotų pažeidžiamumų būsenos

4.3.2. Statinė analizė

Siekiant patikrinti, ar *UI/Gateway WEB* serveris yra sukonfigūruotas teisingai ir saugiai, buvo atliekama statinė konfigūracijos atitikties (angl. *compliance*) analizė. Analizė buvo atliekama pasitelkus *Chef InSpec* įrankį[7] (4.23.15 versiją) ir *https://dev-sec.io/baselines/nginx/* pateiktą testų rinkinį. Pirminės analizės rezultatai yra pateikiami 10 lentelėje.

10 lentelė. Pirminiai konfigūracijos atitikties analizės rezultatai

Profile Summary: 6 successful controls, 8 control failures, 2 controls skipped Test Summary: 15 successful, 17 failures, 2 skipped

Aptikti neatitikimai buvo peržiūrėti, o svarbiausi iš jų - pataisyti:

- Apribotas maksimalus kiekis sujungimų iš vieno *IP* adreso, siekiant apsisaugoti nuo *DOS* atakų;
- Sukonfigūruotos *Content-Security-Policy*, *Strict-Transport-Security* ir *Expect-CT HTTP* antraštės;
- Nurodyti užklausų apdorojimui skirtų buferių maksimalūs dydžiai, siekiant apsisaugoti nuo buferio perpildymo ir *DOS* atakų;
- Sukonfigūruotas *TLS* v1.2 naudojimas.

Pakartotiniai analizės rezultatai, atlikti po pataisymų, yra pateikiami 11 lentelėje.

11 lentelė. Pakartotiniai konfigūracijos atitikties analizės rezultatai

Profile Summary: 11 successful controls, 3 control failures, 2 controls skipped Test Summary: 28 successful, 4 failures, 2 skipped

[7] https://docs.chef.io/inspec/

Likutiniai konfigūracijos neatitikimai:

- Neatitinkantis proceso savininko vardas – teste tikimasi, jog procesas bus vykdomas *www-data* naudotojo teisėmis, tačiau buvo aptiktas *nginx* naudotojas. Tai nėra problema, kadangi *nginx* naudotojo teisės yra apribotos minimaliam veiksmų vykdymui;
- *HTTP Strict-Transport-Security* antraštės neatitikimas – antraštė serverio konfigūracijoje yra nurodyta, tačiau dėl kabučių nesutapimo nebuvo aptikta;
- *HTTP Content-Security-Policy* antraštė – antraštė serverio konfigūracijoje yra nurodyta, tačiau nesutampa su reikšme, kurios tikimasi dėl papildomų saugos parametrų nurodytų šioje antraštėje.

4.4. Įsiskverbimo testavimas

Siekiant užtikrinti aukštą sistemos saugumo lygį, buvo atliekamas atsparumo injekcijoms ir įsiskverbimo testavimas, identifikuojant ir bandant išnaudoti potencialius atakos vektorius.

4.4.1. Atsparumo injekcijoms testavimas

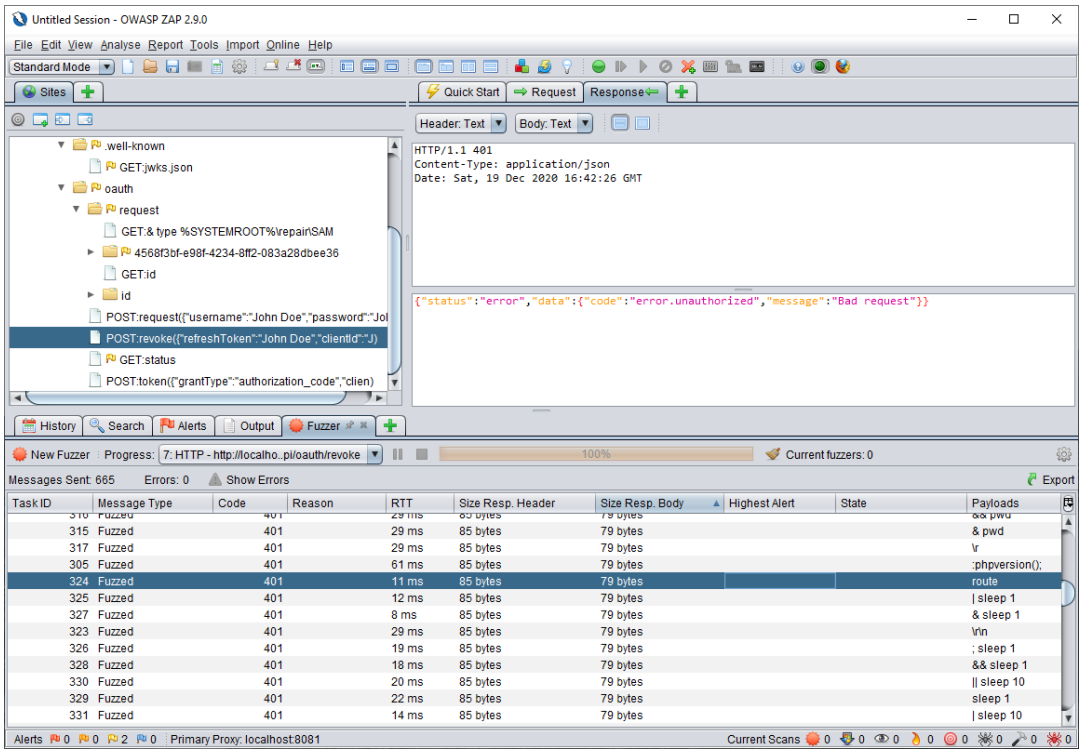
Siekiant užtikrinti, jog neautorizuotiems naudotojams pasiekiami vieši autentifikacijos *API* prieigos taškai yra apsaugoti nuo *SQL* bei *OS* komandų injekcijų, buvo vykdomas įsiskverbimo testavimas į prieigos taškus pateikiant pavojingas neapibrėžtas (angl. *fuzzy*) įvestis.

Testavimas buvo vykdomas vadovaujantis šia metodika:

- Testavimo metu buvo naudojamas *OWASP ZAP* 2.9.0 įrankis ir jo *Fuzzer* modulis;
- Sistemos prototipas buvo paleistas lokalioje aplinkoje pagal 3.7 poskyryje pateikiamas instrukcijas;
- Testavimo metu buvo naudojamas pavojingų neapibrėžtų įvesčių rinkinys[8], kurį sudarė 676 įvestys;
- Iš viso buvo testuojami 6 autentifikacijos *API* prieigos taškai, kurie priima tiek *GET*, tiek *POST* tipo *HTTP* užklausas;
- Pradedant individualaus prieigos taško testavimą, pirmiausiai buvo sudaroma teisinga *HTTP* užklausa ir įsitikinama, jog sistema sėkmingai atlieka nurodytą veiksmą ir grąžina 200 *OK HTTP* statuso kodą;
- Tuomet iš eilės kiekviena prieigos taško įvestis buvo pakeičiama pavojingomis reikšmėmis, siekiant užtikrinti, jog kiekviena prieigos taško įvestis yra testuojama atskirai;
- Paleidus įrankį ir sėkmingai atlikus testą, buvo peržiūrimi rezultatai, kuriuose yra nurodomi serverio grąžinti *HTTP* atsakymai;
- Analizuojant *HTTP* atsakymus, buvo tikimasi, jog bus grąžintas vienas iš numatytų klaidos kodų su klaidos paaiškinimu *HTTP* atsakymo pakete:
    - 400 *BAD REQUEST*;
    - 401 *UNAUTHORIZED*;
    - 404 *NOT FOUND*.
- Kuomet yra patikrinamos visos vieno prieigos taško įvestys, pradedama testuoti sekantį prieigos tašką.

[8] https://github.com/1N3/IntruderPayloads/blob/master/FuzzLists/full_fuzz.txt

77 pav. yra pateikiamas pavyzdinis testavimo rezultatų ekrano vaizdas.



77 pav. *ZAP* įrankio panaudojimas vykdant įsiskverbimo testavimą

Žemiau esančiose lentelėse yra pateikiami apibendrinti prieigos taškų testavimo rezultatai.

12 lentelė. Autentifikacijos užklausos informacijos pateikimo prieigos taško įsiskverbimo testavimo rezultatai

HTTP paketo užklausos metodas (angl. *verb*)	*GET*
URI	*/api/oauth/request/{id}*
HTTP užklausos paketo struktūra	-
Rezultatai	Su tam tikromis įvestimis (pvz. *C:\boot.ini*) buvo grąžintas 404 *HTTP* statuso kodas su *HTML* klaidos tinklalapiu, pateikiamu *HTTP* pakete. Tai yra standartinis *Spring Boot* karkaso veikimas, kuomet nė vienas prieigos taškas negali apdoroti pateiktos *HTTP* užklausos; Su tam tikromis įvestimis *HTTP* pakete buvo grąžintas *JSON* dokumentas, kurio *path* atribute buvo atspindima (angl. *reflected*) netinkama įvestis. Šis funkcionalumas taip pat yra teikiamas *Spring Boot* karkaso. Kadangi *HTTP* atsakymo kūne yra pateikiamas *JSON*, o ne *HTML* dokumentas, ši karkaso ypatybė negali iššaukti *XSS* pažeidžiamumo.

13 lentelė. Autentifikacijos patvirtinimo prieigos taško įsiskverbimo testavimo rezultatai

HTTP paketo užklausos metodas (angl. *verb*)	*POST*
URI	*/api/oauth/request/{id}/approve- authentication*

HTTP užklausos paketo struktūra	*{* *"token": "string"* *}*
Rezultatai	Su visomis įvestimis buvo gautas 400 *HTTP* statuso atsakymas, tačiau atsakymo kūne buvo pateiktas *Internal server error* klaidos pranešimas; Neinformatyvus klaidos pranešimas buvo pakeistas informatyvesniu *Failed to parse token* pranešimu.

14 lentelė. Autorizacijos patvirtinimo prieigos taško įsiskverbimo testavimo rezultatai

HTTP paketo užklausos metodas (angl. *verb*)	*POST*
URI	*/api/oauth/request/{id}/approve-authorization*
HTTP užklausos paketo struktūra	*{* *"token": "string"* *}*
Rezultatai	Su visomis įvestimis buvo gautas 400 *HTTP* statuso atsakymas, tačiau atsakymo kūne buvo pateiktas *Internal server error* klaidos pranešimas; Neinformatyvus klaidos pranešimas buvo pakeistas informatyvesniu *Failed to parse token* pranešimu.

15 lentelė. Autentifikacijos užklausos sukūrimo prieigos taško įsiskverbimo testavimo rezultatai

HTTP paketo užklausos metodas (angl. *verb*)	*POST*
URI	*/api/oauth/request*
HTTP užklausos paketo struktūra	*{* *"username": "string",* *"password": "string",* *"clientId": "string",* *"redirectUrl": "string",* *"codeChallenge": "string"* *}*
Rezultatai	Su tam tikru įvesčių poaibiu *codeChallenge* parametras buvo panaudotas sėkmingai sukuriant autentifikacijos užklausą; Kadangi šis parametras yra naudojamas maišos funkcijos apskaičiavimui ir jų reikšmių sulyginimui, papildoma įvesties validacija nėra būtina ir netinkamos įvestys nesudarys sąlygų injekcijos pažeidžiamumų atsiradimui.

16 lentelė. Atnaujinimo žetono atšaukimo prieigos taško įsiskverbimo testavimo rezultatai

HTTP paketo užklausos metodas (angl. *verb*)	*POST*
URI	*/api/oauth/revoke*
HTTP užklausos paketo struktūra	*{* *"refreshToken": "string",* *"clientId": "string",* *"clientSecret": "string"* *}*
Rezultatai	Nenumatyta elgsena neaptikta.

17 lentelė. Žetonų išdavimo prieigos taško įsiskverbimo testavimo rezultatai

HTTP paketo užklausos metodas (angl. *verb*)	*POST*
URI	*/api/oauth/token*
HTTP užklausos paketo struktūra	*{* *"grantType": "authorization_code",* *"clientId": "string",* *"clientSecret": "string",* *"refreshToken": "string",* *"code": "string",* *"codeVerifier": "string"* *}*
Rezultatai	Nenumatyta elgsena neaptikta.

Testavimo metu aptikta nenumatyta sistemos elgsena buvo ištaisyta arba priskirta kaip nepavojinga ir tenkinanti prototipui keliamus realizacijos reikalavimus.

4.4.2. Įsiskverbimo testavimo scenarijai

Siekiant patikrinti, ar siūlomas autentifikacijos metodas yra atsparus tam tikriems atakos vektoriams, buvo sudaryti įsiskverbimo testavimo scenarijai. Siekiant automatizuoti įsiskverbimo testavimą, buvo sukurta atskira *Spring Boot* testavimo karkasą taikanti programa, parašyta *Java* programavimo kalba. Įsiskverbimo testavimo programa teikia šį funkcionalumą:

- Scenarijams reikalingų duomenų įrašymą į duomenų bazę;
- Kreipimąsi į autentifikacijos *API* prieigos taškus su scenarijuose nurodytomis *HTTP* užklausomis;
- Scenarijų vykdymą ir rezultatų pateikimą.

Įvykdžius testavimo scenarijus, buvo įvertinta, ar numatyti atakos vektoriai gali lemti metodo saugos pažeidžiamumus. 18 lentelėje yra pateikiami sudaryti įsiskverbimo testavimo scenarijai ir jų vykdymo rezultatai. 6 priede yra pateikiamos testų *HTTP* užklausos ir atsakymai.

18 lentelė. Įsiskverbimo testavimo scenarijai ir jų vykdymo rezultatai

Scenarijus Nr. 1	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali naudoti grubios jėgos metodą siekiant atspėti slaptažodį.
Susiję prieigos taškai	*/api/oauth/request*
Automatizuoto testo pavadinimas	*RequestCreationBruteForcePassword*
Rezultatai	Testo metu buvo atliktos 3 užklausos su neteisingu slaptažodžiu, imituojant grubios jėgos ataką. Ketvirtoji užklausa buvo atlikta su teisingu slaptažodžiu, bet sistema grąžino klaidos pranešimą, nurodydama, jog paskyra buvo užblokuota. Toks funkcionalumas yra teisingas ir numatytas 3.8 nefunkciniame reikalavime „Nesėkmingai atlikus 3 bandymus prisijungti prie paskyros, paskyra yra užblokuojama. Užblokuotą paskyrą atblokuoti gali tik administratorius“. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.

Scenarijus Nr. 2	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali suklastoti *redirectUrl* parametrą ir potencialiai po autentifikacijos nukreipti naudotojus į piktavališkus tinklalapius.
Susiję prieigos taškai	*/api/oauth/request*
Automatizuoto testo pavadinimas	*RequestCreationTamperRedirectUrl*
Rezultatai	Testo metu buvo atlikta užklausa su suklastotu *redirectUrl* parametru. Sistema grąžino klaidos pranešimą, nurodydama, jog pateiktas *redirectUrl* parametras yra neteisingas. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 3	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali suklastoti *clientId* parametrą ir bandyti prisijungti prie sistemų, kurių neturėtų galėti pasiekti.
Susiję prieigos taškai	*/api/oauth/request*
Automatizuoto testo pavadinimas	*RequestCreationTamperClientId*
Rezultatai	Testo metu buvo atlikta užklausa su suklastotu *clientId* parametru. Sistema grąžino klaidos pranešimą, nurodydama, jog naudotojas negali gauti prieigos prie sistemos, turinčios pateiktą *clientId* parametrą. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 4	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali suklastoti *JWT* žetoną, kurio parašas nebūtų teisingas.
Susiję prieigos taškai	*/api/oauth/request/{id}/approve-authentication* */api/oauth/request/{id}/approve-authorization*
Automatizuoto testo pavadinimas	*ApproveAuthenticationInvalidSignature* *ApproveAuthorizationInvalidSignature*
Rezultatai	Testo metu buvo atliktos užklausos su žetonais, kurių parašai buvo suklastoti. Sistema grąžino klaidos pranešimus, nurodydama, jog negali sėkmingai apdoroti nurodytų žetonų. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 5	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali suklastoti *JWT* žetoną, antraštėje pakeisdami pasirašymo metodą į *None* ir taip apeidami *JWT* žetono turinio patikrą.
Susiję prieigos taškai	*/api/oauth/request/{id}/approve-authentication* */api/oauth/request/{id}/approve-authorization*
Automatizuoto testo pavadinimas	*ApproveAuthenticationNoneAlgorithm* *ApproveAuthorizationNoneAlgorithm*

Rezultatai	Testo metu buvo atliktos užklausos su žetonais, kurių pasirašymo metodas buvo suklastotas. Sistema pateikė klaidos pranešimus, nurodydama, jog negali sėkmingai apdoroti nurodytų žetonų. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 6	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali suklastoti *JWT* žetone esantį žetono išdavimo laiko atributą, siekdami panaudoti dar negaliojantį žetoną.
Susiję prieigos taškai	*/api/oauth/request/{id}/approve-authentication* */api/oauth/request/{id}/approve-authorization*
Automatizuoto testo pavadinimas	*ApproveAuthenticationTamperNotBefore* *ApproveAuthorizationTamperNotBefore*
Rezultatai	Testo metu buvo atliktos užklausos su žetonais, kurių panaudojimo pradžios laikas buvo suklastotas ir nurodytas į ateitį. Sistema pateikė klaidos pranešimus, nurodydama, jog negali sėkmingai apdoroti nurodytų žetonų. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 7	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali suklastoti *JWT* žetone esantį naudotojo identifikatoriaus atributą, siekdami patvirtinti užklausas kitiems naudotojams, kurie nėra jų pavaldiniai.
Susiję prieigos taškai	*/api/oauth/request/{id}/approve-authorization*
Automatizuoto testo pavadinimas	*ApproveAuthorizationApproveOtherRequest*
Rezultatai	Testo metu buvo atlikta užklausa su žetonu, kuriame buvo nurodyta, jog norima autorizuoti prieigą naudotojui, kuris nėra nurodytas kaip pavaldinys. Sistema pateikė klaidos pranešimą, nurodydama, jog negali sėkmingai įvykdyti nurodytos užklausos. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 8	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali suklastoti *JWT* žetone esantį naudotojo identifikatoriaus atributą, siekdami patvirtinti užklausą kito naudotojo vardu.
Susiję prieigos taškai	*/api/oauth/request/{id}/approve-authentication*
Automatizuoto testo pavadinimas	*ApproveAuthenticationApproveOtherRequest*
Rezultatai	Testo metu buvo atlikta užklausa su žetonu, kuris buvo sukurtas kito naudotojo vardu. Pirminio testavimo metu sistema sėkmingai priėmė suklastotą žetoną ir patvirtino autentifikaciją. Aptiktas saugos pažeidžiamumas buvo ištaisytas ir pakartotinio testavimo metu sistema pateikė klaidos pranešimą, nurodydama, jog negali sėkmingai įvykdyti nurodytos užklausos. Išvada – ištaisius aptiktą saugos pažeidžiamumą, sistema tapo apsaugota nuo identifikuoto atakos vektoriaus.

Scenarijus Nr. 9	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali suklastoti *JWT* žetone esantį žetono galiojimo pabaigos laiko atributą, siekdami panaudoti jau nebegaliojantį žetoną.
Susiję prieigos taškai	*/api/oauth/request/{id}/approve-authentication* */api/oauth/request/{id}/approve-authorization*
Automatizuoto testo pavadinimas	*ApproveAuthenticationTamperExpiration* *ApproveAuthorizationTamperExpiration*
Rezultatai	Testo metu buvo atliktos užklausos su žetonais, kurių galiojimo laikas buvo pasibaigęs. Sistema pateikė klaidos pranešimus, nurodydama, jog negali sėkmingai apdoroti nurodytų žetonų. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 10	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali suklastoti *JWT* žetone esantį veiksmo atlikimo atributą (pvz. patvirtinti autorizaciją), siekdami atlikti veiksmą, kuriam neturi įgaliojimų.
Susiję prieigos taškai	*/api/oauth/request/{id}/approve-authentication* */api/oauth/request/{id}/approve-authorization*
Automatizuoto testo pavadinimas	*ApproveAuthenticationTamperAction* *ApproveAuthorizationTamperAction*
Rezultatai	Testo metu buvo sudarytos užklausos, kurių žetonuose vadovas bandė patvirtinti autentifikacijos užklausą, o naudotojas bandė autorizuoti užklausą. Sistema pateikė klaidos pranešimus, nurodydama, jog negali sėkmingai įvykdyti nurodytų užklausų. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 11	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali atlikti pakartojimo ataką, siekdami išnaudoti perimtą teisėtos užklausos informaciją.
Susiję prieigos taškai	*/api/oauth/request/{id}/approve-authentication* */api/oauth/request/{id}/approve-authorization*
Automatizuoto testo pavadinimas	*ApproveAuthenticationReplay* *ApproveAuthorizationReplay*
Rezultatai	Testo metu patvirtinimo užklausos buvo pakartojamos, imituojant perimtų duomenų pakartojimo ataką. Vykdant pakartotines užklausas, sistema pateikė klaidos pranešimus, nurodydama, jog negali sėkmingai įvykdyti nurodytų užklausų. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 12	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali pateikti teisėtą užklausą, kuomet autentifikacijos proceso sesija yra pasibaigusi.

Susiję prieigos taškai	*/api/oauth/request/{id}/approve-authentication* */api/oauth/request/{id}/approve-authorization*
Automatizuoto testo pavadinimas	*ApproveAuthenticationExpiredSession* *ApproveAuthorizationExpiredSession*
Rezultatai	Testo metu buvo sukurtos autentifikacijos sesijos, kurios prieš siunčiant patvirtinimo žetonus duomenų bazėje buvo nustatytos kaip pasibaigusios. Gavusi žetonus, sistema pateikė klaidos pranešimus, nurodydama, jog žetonas negali būti priimtas, kadangi autentifikacijos sesija yra pasibaigusi. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 13	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali panaudoti pasibaigusį atnaujinimo žetoną, norėdami gauti naują prieigos žetoną ir įgauti neribotą prieigos sesiją.
Susiję prieigos taškai	*/api/oauth/token*
Automatizuoto testo pavadinimas	*ExchangeTokenExpiredRefreshToken*
Rezultatai	Testo metu pirmiausiai buvo gautas galiojantis atnaujinimo žetonas. Tuomet duomenų bazėje šis žetonas buvo nustatytas kaip nebegaliojantis. Vykdant žetono atnaujinimo užklausą, sistema pateikė klaidos pranešimą, nurodydama, jog negali įvykdyti užklausos su neteisingu atnaujinimo žetonu. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 14	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali suklastoti *clientId* parametrą ir bandyti gauti žetonus, skirtus prieigai prie sistemų, kurių neturėtų galėti pasiekti.
Susiję prieigos taškai	*/api/oauth/token*
Automatizuoto testo pavadinimas	*ExchangeTokenTamperClientId*
Rezultatai	Testo metu buvo atlikta užklausa su suklastotu *clientId* parametru. Pirminio testavimo metu sistema sėkmingai priėmė suklastotą užklausą ir suteikė prieigos žetoną prie sistemos, kurios naudotojas neturėtų pasiekti. Aptiktas saugos pažeidžiamumas buvo ištaisytas ir pakartotinio testavimo metu sistema pateikė klaidos pranešimą, nurodydama, jog negali sėkmingai įvykdyti nurodytos užklausos. Išvada – ištaisius aptiktą saugos pažeidžiamumą, sistema tapo apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 15	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali suklastoti *clientSecret* parametrą ir bandyti gauti žetonus, skirtus prieigai prie sistemų, kurių neturėtų galėti pasiekti.
Susiję prieigos taškai	*/api/oauth/token*
Automatizuoto testo pavadinimas	*ExchangeTokenTamperClientSecret*

Rezultatai	Testo metu buvo atlikta užklausa su suklastotu *clientSecret* parametru. Sistema grąžino klaidos pranešimą, nurodydama, jog susijęs *OAuth 2.0* klientas nebuvo rastas. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 16	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali suklastoti *codeVerifier* parametrą ir bandyti apeiti *PKCE* saugos mechanizmą.
Susiję prieigos taškai	*/api/oauth/token*
Automatizuoto testo pavadinimas	*ExchangeTokenTamperCodeVerifier*
Rezultatai	Testo metu buvo atlikta užklausa su suklastotu *codeVerifier* parametru. Sistema grąžino klaidos pranešimą, nurodydama, jog pateiktas *codeVerifier* parametras yra neteisingas. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.
Scenarijus Nr. 17	
Atakos vektorius	Vykdant užklausas į prieigos tašką, piktavaliai gali atlikti pakartojimo ataką, siekdami išnaudoti perimtą teisėtos užklausos informaciją.
Susiję prieigos taškai	*/api/oauth/token*
Automatizuoto testo pavadinimas	*ExchangeTokenTamperReplay*
Rezultatai	Testo metu žetonų gavimo užklausa buvo pakartojama, imituojant perimtų duomenų pakartojimo ataką. Vykdant pakartotinę užklausą, sistema pateikė klaidos pranešimą, nurodydama, jog susijęs autorizacijos kodas nėra rastas, t.y. jau buvo panaudotas. Išvada – sistema yra apsaugota nuo identifikuoto atakos vektoriaus.

4.5. Prieigos taškų greitaveikos testavimas

Siekiant patikrinti, ar sukurta sistema tenkina greitaveikai keliamą 1.8 nefunkcinį reikalavimą, teigiantį, jog „*API* prieigos taškai naudotojų užklausas turi apdoroti greičiau nei per 500 milisekundžių, kuomet sistema lygiagrečiai naudojasi 100 naudotojų“, buvo atliekamas greitaveikos testavimas. Šio testavimo metu buvo siekiama įvertinti autentifikacijos *API* prieigos taškų atsako laikus ir ištirti, ar *API* programa gali užtikrinti stabilų darbą, kuomet ja naudojasi reikalavimuose nurodytas kiekis lygiagrečių naudotojų.

4.5.1. Testavimo duomenų paruošimas

Eksperimento atlikimui yra svarbu užtikrinti kuo artimesnę realiam sistemos naudojimui skirtą aplinką, todėl turi būti sukurta automatizuota priemonė, leidžianti pripildyti duomenų bazę nurodytais kiekiais sugeneruotų duomenų.

Šiam tikslui pasiekti buvo sukurta atskira *Spring Boot* karkasą taikanti programa, parašyta *Java* programavimo kalba. Duomenų generavimo programa teikia šį funkcionalumą:

– Konfigūracijos faile nurodoma kiek naudotojų paskyrų, *OAuth* 2.0 klientų bei istorinių autentifikacijos užklausų reikia sugeneruoti;
– Paleidus programą, nurodyti duomenų kiekiai yra sugeneruojami ir įrašomi į duomenų bazę;
– Paleidus programą, yra pateikiami du *API* prieigos taškai, skirti autentifikacijos ir autorizacijos užklausų patvirtinimo žetonų generavimui. Šie prieigos taškai yra skirti dinamiškai generuoti duomenis, kuriuos sukurtų taikomoji mobiliojo ryšio programa.

Eksperimentas buvo vykdomas naudojant skirtingų apimčių duomenų rinkinius, siekiant įvertinti sistemos greitaveikos priklausomybę nuo sistemoje esančių duomenų kiekio. Kadangi duomenys apie realių kritinės infrastruktūros sistemų apimtis nėra teikiami, eksperimento tikslais buvo sudaryti hipotetinės apimties duomenų rinkiniai. Eksperimento metu naudojamų rinkinių konfigūracijos yra pateikiamos 19 lentelėje.

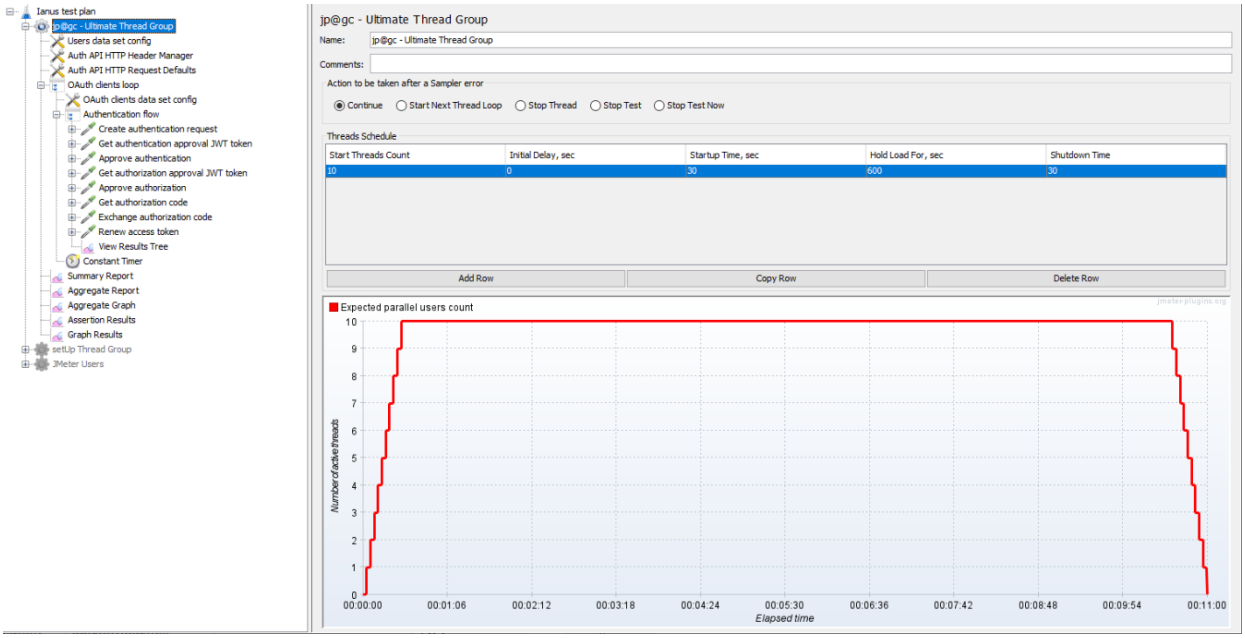
19 lentelė. Greitaveikos eksperimento duomenų rinkinių konfigūracijos

Duomenų esybė	**Rinkinys #1**	**Rinkinys #2**	**Rinkinys #3**	**Rinkinys #4**	**Rinkinys #5**
Lygiagrečių naudotojų	10	20	50	100	200
Naudotojų paskyrų sistemoje	10	100	500	1000	2000
OAuth 2.0 klientų	3	10	25	50	100
Istorinių autentifikacijos užklausų	3000	30000	150000	300000	600000

4.5.2. Scenarijaus paruošimas

Eksperimentas buvo vykdomas pasitelkiant *JMeter* įrankį, kuris lygiagrečiai vykdo *HTTP* užklausas į prieigos taškus. Lygiagretumo valdymui buvo pasitelktas *Ultimate Thread Group* papildinys, kuris leidžia nurodyti kiek lygiagrečių scenarijų reikia vykdyti, kokie yra palaipsninės pradžios ir pabaigos (angl. *ramp-up* ir *ramp-down*) periodai bei kiek laiko turi būti vykdomas eksperimentas.

78 pav. yra pateiktas *JMeter* įrankio ekrano vaizdas ir *Ultimate Thread Group* papildinio nustatymai, skirti šiam eksperimentui – 30 sekundžių palaipsninės pradžios/pabaigos periodai ir 10 minučių viso eksperimento vykdymui.



78 pav. *JMeter* įrankio ekrano vaizdas

Testo scenarijų sudaro keletas žingsnių, kurių informacija yra pateikiama 20 lentelėje.

20 lentelė. Greitaveikos testavimo scenarijaus žingsniai

Nr.	Pavadinimas	Komentaras	Prieigos taškas	Programa
1.	*Create authentication request*	Pradedamas autentifikacijos procesas	*/api/oauth/request*	Autentifikacijos *API*
2.	*Get authentication approval JWT token*	Sugeneruojamas žetonas, skirtas patvirtinti autentifikaciją panaudojant antrąjį faktorių	*/api/approvals/${username}/authentication*	Duomenų generavimo *API*
3.	*Approve authentication*	Užklausa patvirtinama panaudojant sugeneruotą autentifikacijos patvirtinimo žetoną	*/api/oauth/request/${requestId}/approve-authentication*	Autentifikacijos *API*
4.	*Get authorization approval JWT token*	Sugeneruojamas žetonas, skirtas patvirtinti autorizaciją	*/api/approvals/${username}/authorization*	Duomenų generavimo *API*
5.	*Approve authorization*	Užklausa patvirtinama panaudojant sugeneruotą autorizacijos patvirtinimo žetoną	*/api/oauth/request/${requestId}/approve-authorization*	Autentifikacijos *API*
6.	*Get authorization code*	Iš patvirtintos autentifikacijos užklausos yra gaunamas autorizacijos kodas	*/api/oauth/request/${requestId}*	Autentifikacijos *API*
7.	*Exchange authorization code*	Autorizacijos kodas yra iškeičiamas į prieigos ir atnaujinimo žetonus	*/api/oauth/token*	Autentifikacijos *API*
8.	*Renew access token*	Atnaujinimo žetonas yra iškeičiamas į naują prieigos žetoną	*/api/oauth/token*	Autentifikacijos *API*

Testavimo scenarijus yra išsaugomas *.jmx* faile ir yra paleidžiamas vykdymui naudojant komandą:

jmeter -n –t load_testing.jmx -l results.jtl -e –o reporting

Sėkmingai įvykdžius eksperimentą, rezultatai yra pateikiami *reporting/index.html* faile.

4.5.3. Eksperimento vykdymo metodika

Eksperimentas buvo vykdomas pasitelkiant *AWS* debesijos tiekėjo siūlomus resursus ir paslaugas. Eksperimento vykdymui buvo sukurta tokios konfigūracijos infrastruktūra:

- Vienas *db.t3.medium* tipo *PostgreSQL 12.2-R1 RDS* serveris;
- Vienas *t2.medium* (2 *vCPU, 4GB RAM*) tipo *EC2* serveris, skirtas duomenų generavimo *API* programai;
- Vienas *t2.medium* (2 *vCPU, 4GB RAM*) tipo *EC2* serveris, skirtas autentifikacijos *API* programai.

Eksperimentas buvo vykdomas pagal šią metodiką:

- *AWS* debesijos platformoje sukuriama nurodytos konfigūracijos infrastruktūra;
- *RDS* serveryje sukuriama duomenų bazės schema;
- Į *EC2* serverį įdiegiama ir paleidžiama autentifikacijos *API* programa;
- Į *EC2* serverį įdiegiama duomenų generavimo *API* programa;
- Nurodoma generuojamų duomenų rinkinio konfigūracija ir paleidžiama duomenų generavimo programa;
- Prisijungus prie *RDS* įsitikinama, jog duomenų rinkinys buvo sugeneruotas sėkmingai;
- Į *EC2* serverį, kuriame buvo įdiegta duomenų generavimo *API* programa, įdiegiamas ir sukonfigūruojamas *JMeter* įrankis;
- *JMeter* įrankis paleidžiamas su nurodyta duomenų rinkinio konfigūracija;
- Sėkmingai užbaigus testą, iš *reporting/index.html* failo surenkami eksperimento metu gauti rezultatai;
- Eksperimentas kartojamas naudojant kitą duomenų rinkinio konfigūraciją.

4.5.4. Eksperimento rezultatai

Eksperimentas buvo vykdomas 5 kartus su skirtingomis duomenų rinkinių konfigūracijomis, nurodytomis 19 lentelėje. 21 lentelėje yra nurodyti eksperimentų vykdymo laikai, padedantys nustatyti infrastruktūros metrikas konkrečiu eksperimento vykdymo metu.

21 lentelė. Eksperimentų vykdymo laikai

Duomenų rinkinio Nr.	**Pradžios laikas (UTC)**	**Pabaigos laikas (UTC)**
#1	12:27:35	12:38:35
#2	12:47:55	12:58:54
#3	13:10:54	13:21:53
#4	13:35:14	13:46:13
#5	14:32:31	14:43:31

22 lentelėje yra pateikiami eksperimento su duomenų rinkiniu #1 rezultatai. Kitų duomenų rinkinių rezultatai yra pateikiami 5 priede.

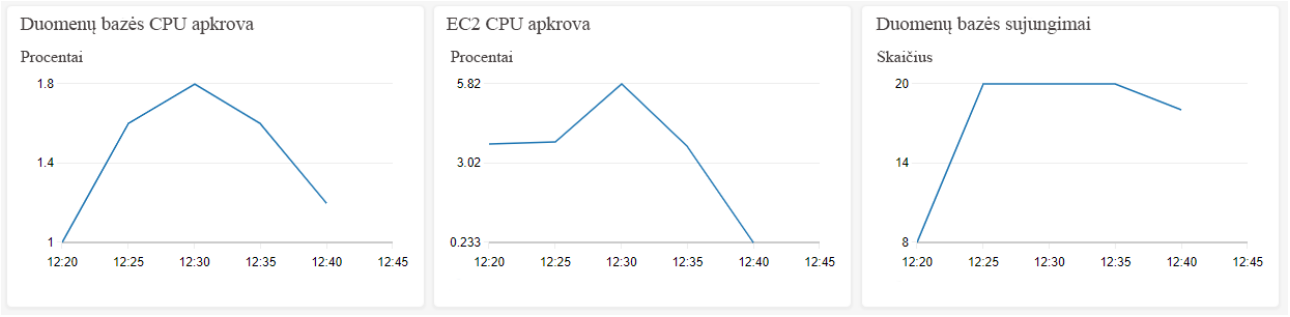
22 lentelė. Eksperimento naudojant duomenų rinkinį #1 greitaveikos rezultatai

Prieigos taško Nr.	Imtis	Vidurkis (ms)	Minimumas (ms)	Maksimumas (ms)	Mediana (ms)	90 procentilis (ms)	95 procentilis (ms)
1.	390	110.85	103	763	107	115	119
2.	390	11.32	5	473	8	18	22
3.	390	14.07	8	108	12	21	25
4.	390	9.34	4	48	7	16	21.45
5.	390	12.92	8	41	11	18	21.45
6.	390	5.27	2	48	4	8	10
7.	389	14.71	8	115	13	21	24
8.	380	7.94	4	32	7	11	13

79 pav. yra pateiktos *AWS CloudWatch* paslaugos siūlomos infrastruktūros metrikos, kuomet buvo vykdomas eksperimentas su duomenų rinkiniu #1. Metrikose yra nurodoma:

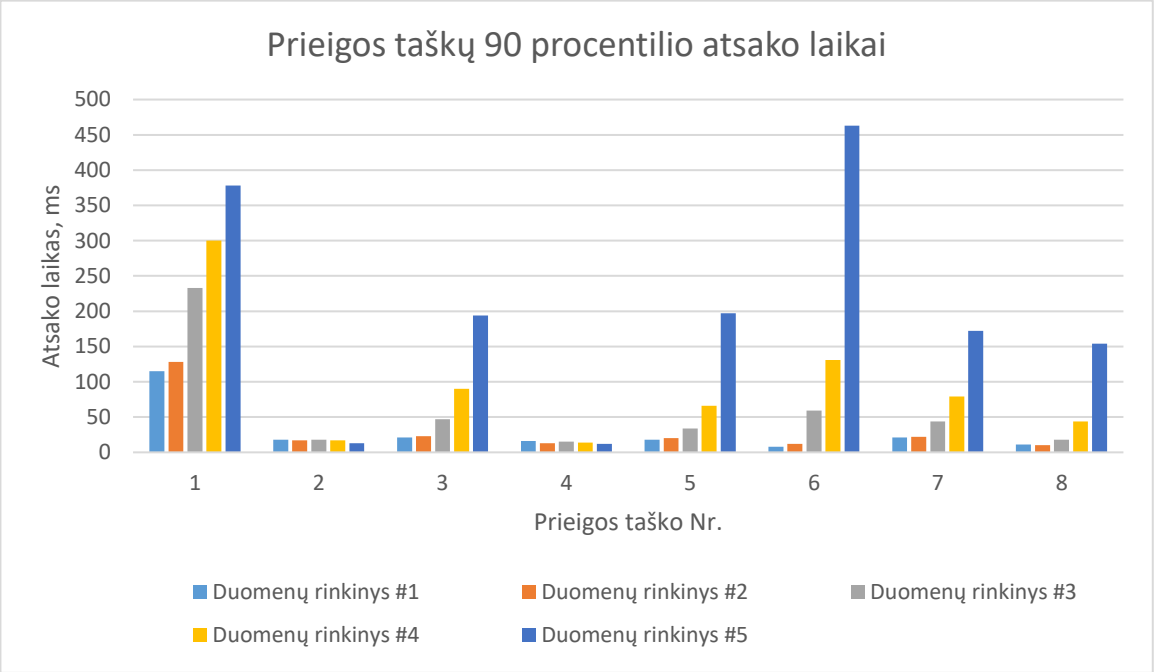
– Duomenų bazės procesoriaus apkrova;
– *EC2* serverio, kuriame veikia autentifikacijos *API* programa, procesoriaus apkrova;
– Duomenų bazės sujungimų skaičius.

Infrastruktūros metrikos, kuomet buvo vykdomi eksperimentai su kitais duomenų rinkiniais, yra pateikiamos 5 priede.



79 pav. Infrastruktūros metrikos, vykdant eksperimentą su duomenų rinkiniu #1

Grafike, kuris pateikiamas 80 pav., yra nurodomi apibendrinti prieigos taškų, kurių paskirtys pateiktos 20 lentelėje, 90 procentilio atsako laikai su skirtingais duomenų rinkiniais.



80 pav. Autentifikacijos *API* prieigos taškų 90 procentilio atsako laikai

Atlikus eksperimentus, surinkus ir išanalizavus rezultatus, buvo nustatyti dėsningumai:

- Vykdant eksperimentus, duomenų bazės sujungimų skaičius nekito ir buvo lygus 20. Šis skaičius yra fiksuotas, kadangi autentifikacijos *API* programa yra sukonfigūruota naudoti fiksuotą duomenų bazės sujungimų telkinį (angl. *connection pool*);
- Didėjant duomenų rinkinių apimtims, tendencingai didėjo duomenų bazės ir *EC2* serverio procesoriaus apkrova. Su didžiausiu #5 duomenų rinkiniu duomenų bazės apkrova pasiekė 70.4% piką, o *EC2* serverio apkrova pasiekė 66.2% piką;
- Remiantis grafiko, kuris pateikiamas 80 pav., duomenimis, didžiausius atsako laikus pateikė pirmasis prieigos taškas (*Create authentication request*);
- Remiantis grafiko, kuris pateikiamas 80 pav., duomenimis, ryškiausias atsako laikų pokytis buvo tarp #4 ir #5 duomenų rinkinių. Didžiosios dalies prieigos taškų atsako laikai padidėjo kelis kartus;
- Su #4 duomenų rinkiniu, kuriame yra naudojama 100 lygiagrečių naudotojų, visų prieigos taškų atsako laikai neperkopė 300 ms. ribos. Šis rezultatas nurodo, jog yra tenkinamas nefunkcinis greitaveikos reikalavimas, teigiantis, jog „*API* prieigos taškai naudotojų užklausas turi apdoroti greičiau nei per 500 milisekundžių, kuomet sistema lygiagrečiai naudojasi 100 naudotojų“;
- Su didžiausiu #5 duomenų rinkiniu, kuriame yra naudojama 200 lygiagrečių naudotojų, didžiausią atsako laiką pateikęs prieigos taškas užklausas apdorojo per 462.9 ms.

4.6. Konfigūracijos ir autentifikavimo procesų atlikimo greičio tyrimas

Siekiant įvertinti, kaip greitai naudotojai gali sukonfigūruoti autentifikavimo metodą ir sėkmingai atlikti patį autentifikacijos procesą, buvo atliekamas šių veiklų atlikimo greičio tyrimas.

Eksperimento metu buvo matuojama kiek laiko naudotojui užtrunka atlikti šias veiklas:

- Susieti mobiliojo ryšio įrenginį;
- Atlikti autentifikacijos procesą be antrojo faktoriaus panaudojimo ir be vadovo autorizacijos;
- Atlikti autentifikacijos procesą su antrojo faktoriaus panaudojimu ir be vadovo autorizacijos;
- Atlikti autentifikacijos procesą su antrojo faktoriaus panaudojimu ir su vadovo autorizacija.

Gautieji rezultatai buvo palyginti su [72] literatūros šaltinyje pateikiamais eksperimentų rezultatais, kurie buvo gauti panaudojant įvairius dviejų faktorių autentifikacijos metodus.

4.6.1. Eksperimento vykdymo metodika

Eksperimentas buvo vykdomas remiantis kuo artimesne metodika, nurodyta [72] literatūros šaltinyje:

- Siekiant užtikrinti tikslias laiko žymes, *UI* programa į naršyklės terminalą išves proceso pradžios ir pabaigos laikus milisekundžių tikslumu;
- Autentifikacijos proceso atlikimo metu bus prisijungiama prie paskyrų valdymo sistemos;
- Viso proceso trukme yra laikomas proceso pabaigos laiko ir proceso pradžios laiko žymių skirtumas;
- Autentifikacijos proceso pradžia yra laikomas momentas, kuomet naudotojas pateikia naudotojo vardo ir slaptažodžio formą;
- Autentifikacijos proceso pabaiga yra laikomas momentas, kuomet naudotojas yra nukreipiamas į paskyrų valdymo sistemos pagrindinį langą;
- Mobiliojo ryšio įrenginio susiejimo proceso pradžia yra laikomas momentas, kuomet naudotojas naršyklėje paspaudžia įrenginio susiejimo mygtuką;
- Mobiliojo ryšio įrenginio susiejimo proceso pabaiga yra laikomas momentas, kuomet naudotojui naršyklėje yra pateikiamas pranešimas apie sėkmingai atliktą susiejimą;
- Matuojant autentifikacijos proceso atlikimo laiką, į jį nėra įtraukiamas laikas, skirtas naudotojui surasti susietą mobiliojo ryšio įrenginį. Daroma prielaida, jog naudotojas visus reikalingus įrenginius iš anksto yra pasiruošęs;
- Aukščiau pateikta prielaida galioja ir vykdant eksperimentą, kuomet į autentifikacijos procesą yra įtraukiama vadovo autorizacija. Daroma prielaida, jog vadovas yra iš anksto pasiruošęs patvirtinti užklausą;
- Kiekvieno proceso atlikimui buvo skiriama 10 bandymų.

4.6.2. Eksperimento rezultatai

23 lentelėje yra pateikiami metodo konfigūracijos proceso atlikimo eksperimentų rezultatai.

23 lentelė. Metodo konfigūracijos proceso atlikimo laiko rezultatai

Metodo pavadinimas	**Mediana (sekundės)**	**Vidurkis (sekundės)**
Iš anksto sugeneruoti kodai	1.0	2.2
Mobilieji pranešimai (angl. *push notifications*)	23.5	27.3
SMS žinutės	32.0	34.5
Laiku paremti vienkartiniai slaptažodžiai (*TOTP*)	84.0	109.6

Metodo pavadinimas	Mediana (sekundės)	Vidurkis (sekundės)
U2F	44.0	57.8
Siūlomas metodas	30.1	29.6

24 lentelėje yra pateikiami autentifikacijos proceso atlikimo eksperimentų rezultatai.

24 lentelė. Autentifikacijos proceso atlikimo laiko rezultatai

Metodo pavadinimas	Mediana (sekundės)	Vidurkis (sekundės)
Iš anksto sugeneruoti kodai	17.2	28.0
Mobilieji pranešimai (angl. *push notifications*)	11.8	16.1
SMS žinutės	16.6	18.5
Laiku paremti vienkartiniai slaptažodžiai (*TOTP*)	15.1	23.9
U2F	9.1	13.0
Siūlomas metodas be antrojo faktoriaus panaudojimo ir be vadovo autorizacijos	1.8	1.9
Siūlomas metodas su antrojo faktoriaus panaudojimu ir be vadovo autorizacijos	12.2	12.3
Siūlomas metodas su antrojo faktoriaus panaudojimu ir su vadovo autorizacija	22.1	21.3

Atsižvelgus į gautų rezultatų medianas, galima teigti, jog siūlomo metodo konfigūracijos proceso atlikimo greitis yra lėtesnis už [72] literatūros šaltinyje nurodytus mobiliųjų pranešimų ir iš anksto sugeneruotų slaptažodžių metodus, tačiau sparta lenkia *SMS* žinučių, laiku paremtų vienkartinių slaptažodžių ir *U2F* metodus.

Tuo tarpu autentifikacijos proceso, kuomet yra naudojamas siūlomas metodas su antrojo faktoriaus panaudojimu ir su vadovo autorizacija, vykdymas užtruko ilgiausiai – apie 22,1 sekundes. Šį rezultatą lemia tai, jog vieno autentifikacijos proceso atlikimo metu naudotojai turi patvirtinti du pranešimus, ateinančius į jų susietus mobiliojo ryšio įrenginius. To pasekoje matomas dėsningumas, jog siūlomo metodo atlikimo laikas yra artimas dvigubam [72] literatūros šaltinyje nurodytam mobiliųjų pranešimų metodo atlikimo laikui.

4.7. Išvados

Statinės kodo analizės priemonės *SonarQube* panaudojimas prototipo komponentų analizei leido aptikti ir išspręsti 170 kodo spragų. Analizės atlikimas taip pat leido apsisaugoti nuo galimų saugumo rizikų naudojant *SHA256* santraukos funkciją, realizuojant apsaugą nuo *Cross-site request forgery* tipo atakų ir apsaugant *API* prieigos taškus taikant *Spring Security* karkasą.

Remiantis [65] literatūros šaltinyje pasiūlytu autentifikavimo metodų vertinimo karkasu ir pritaikius skaitinį charakteristikų įgyvendinimo įvertinimo metodą, siūlomas autentifikavimo metodas buvo įvertintas aukščiausiai (24 balai iš 30,5), lyginant su kitais tirtais metodais. Iš to galima daryti išvadą, kad siūlomas autentifikavimo metodas yra kokybiškesnis panaudojamumo, dislokavimo ir saugumo kategorijų visumos atžvilgiu.

WEB serverių konfigūracijos dinaminė ir statinė analizė leido aptikti ir išspręsti dažniausiai pasitaikančias serverių konfigūracijos problemas ir apsisaugoti nuo pavojingų *HTTP* užklausų, kurios galėtų žalingai paveikti sistemą.

Įsiskverbimo testavimo metu taikytas atsparumo injekcijoms testavimas leido įsitikinti, jog neautorizuotiems naudotojams viešai prieinami sistemos prieigos taškai yra apsaugoti nuo pavojingų įvesčių, galinčių lemti *SQL* ir *OS* komandų injekcijų pažeidžiamumus.

Potencialių atakos vektorių identifikavimas ir įsiskverbimo testavimo scenarijų sudarymas bei įvykdymas padėjo aptikti ir ištaisyti kelis užklausų klastojimo, apsimetinėjimo ir netinkamo prieigos kontrolės valdymo saugos pažeidžiamumus, kurie galėjo sukompromituoti autentifikacijos metodo saugų veikimą.

Sistemos prieigos taškų greitaveikos testavimas leido nustatyti bei įvertinti prieigos taškų atsako į užklausas laikus, taip garantuojant, kad *API* programa gali užtikrinti stabilų ir greitą darbą, kuomet ja naudojasi nefunkciniuose reikalavimuose nurodytas kiekis lygiagrečių naudotojų.

Autentifikacijos metodo konfigūracijos ir autentifikavimo procesų atlikimo greičio tyrimo metu įvertinta, kad naudotojai metodą gali sukonfigūruoti per 30,1 sekundę, o autentifikacijos procesą atlikti per 22,1 sekundes.

Tyrimo metu šių procesų atlikimo sparta taip pat buvo palyginta su kitais autentifikavimo metodais. Atsižvelgus į gautų rezultatų medianas, gauta išvada, jog siūlomo metodo konfigūracijos proceso atlikimo greitis (30,1 sekundė) yra lėtesnis už [72] literatūros šaltinyje nurodytus mobiliųjų pranešimų (23,5 sekundės) ir iš anksto sugeneruotų slaptažodžių (1 sekundė) metodus, tačiau sparta lenkia *SMS* žinučių (32 sekundės), laiku paremtų vienkartinių slaptažodžių (84 sekundės) ir *U2F* (44 sekundės) metodus.

Tuo tarpu autentifikacijos proceso, kuomet yra naudojamas siūlomas metodas su antrojo faktoriaus panaudojimu ir su vadovo autorizacija, vykdymas užtruko ilgiausiai. Šį rezultatą lėmė tai, jog vieno autentifikacijos proceso atlikimo metu naudotojai turi patvirtinti du pranešimus, ateinančius į jų susietus mobiliojo ryšio įrenginius. To pasekoje nustatytas dėsningumas, jog siūlomo metodo atlikimo laikas yra artimas dvigubam [72] literatūros šaltinyje nurodytam mobiliųjų pranešimų metodo atlikimo laikui.

Išvados

1. Išanalizavus kritinės infrastruktūros sistemose naudojamą autentifikacijos procesą ir jam keliamus saugos reikalavimus, buvo identifikuoti trys svarbiausi reikalavimai: 1) turi būti naudojamas stiprus kelių faktorių autentifikacijos metodas; 2) turi būti nustatyta procedūra, suteikianti galimybę aukštesnio rango darbuotojui patvirtinti nuotolinės prieigos prašymą; 3) turi būti užtikrinta, kad nuotolinė prieiga ir autentifikacijos procesas vyktų naudojant saugų informacijos perdavimo kanalą.
2. Išanalizavus autentifikacijos metodus, esančius *Kubernetes* konteinerių klasterio valdymo skydelyje, buvo nustatyta, kad valdymo skydelio apsauga yra neatitinkanti kibernetinio saugumo reikalavimų kritinės infrastruktūros sistemoms, todėl yra reikalinga sukurti bei pritaikyti papildomus saugos mechanizmus, kurie leistų įgyvendinti keliamus saugos reikalavimus.
3. Atlikus egzistuojančių autentifikacijos metodų bei juos naudojančių paslaugų analizę, buvo nustatyta, jog nė vienas iš analizuotų metodų netenkina visų trijų kritinės infrastruktūros sistemų saugos reikalavimų, keliamų autentifikacijos mechanizmui, tad buvo patvirtintas poreikis pasiūlyti naują metodą, kuris realizuotų šiuos reikalavimus.
4. Suprojektuotas autentifikacijos metodas tenkina: 1) kritinės infrastruktūros sistemoms keliamą kelių kanalų (angl. *out-of-band authentication*) panaudojimo reikalavimą, kadangi yra naudojama tiek naudotojo kompiuteryje veikianti naršyklė, tiek naudotojo mobiliojo ryšio įrenginys; 2) autentifikacijos užklausos patvirtinimo (autorizacijos) reikalavimą, kadangi naudotojui nuotolinė prieiga yra suteikiama tik tuo atveju, jei nuotolinės prieigos prašymo užklausą patvirtina jo vadovas; 3) trečiąjį reikalavimą - sistemos komponentai tiek tarpusavyje, tiek su naudotojo įrenginiais komunikuoja naudodami patikimą informacijos perdavimo kanalą: *HTTPS* protokolą.
5. Autorizacijos proceso įgyvendinimui pasirinktas *OAuth 2.0* protokolas leido užtikrinti siūlomo metodo realizavimą suprojektuotos vieningo prisijungimo sistemos prototipe, skirtame subjekto prisijungimui ir identitetui valdyti.
6. Tarpinės įgaliotojo serverio sistemos realizacija ir integracija su vieningo prisijungimo sistema leido apsaugoti nuotolinę prieigą prie *Kubernetes* valdymo skydelio, naudojant suprojektuotą autentifikavimo metodą.
7. Atlikus eksperimentinį pasiūlyto metodo veikimo tyrimą, buvo nustatyta, kad metodą realizuojanti sistema yra apsaugota nuo pavojingų *HTTP* užklausų, o viešai prieinami sistemos prieigos taškai yra apsaugoti nuo pavojingų įvesčių, galinčių lemti *SQL* ir *OS* komandų injekcijų pažeidžiamumus; autentifikacijos *API* programos realizacija užtikrina stabilų ir greitą darbą pagal jai keliamą nefunkcinį greitaveikos reikalavimą (<500 ms 100 naudotojų); metodo konfigūracijos proceso atlikimo greitis yra vidutinis (30,1 s), o autentifikacijos proceso atlikimas užtrunka ilgiausiai (22,1 s), tačiau metodas yra kokybiškesnis panaudojamumo, dislokavimo ir saugumo kategorijų visumos atžvilgiu (24 balai iš 30,5), lyginant jį su kitais tirtais metodais.

Literatūros sąrašas

1. What is Information Infrastructure | IGI Global. In [interaktyvus]. [žiūrėta 2019-12-12]. Prieiga per internetą: <https://www.igi-global.com/dictionary/improving-broadband-access-rural-areas/14415>.

2. 818 Dėl Lietuvos Respublikos kibernetinio saugumo įstatymo įgyvendinimo. In [interaktyvus]. [žiūrėta 2019-12-12]. Prieiga per internetą: <https://e-seimas.lrs.lt/portal/legalAct/lt/TAD/94365031a53411e8aa33fe8f0fea665f/asr>.

3. ALCARAZ, C. - ZEADALLY, S. Critical infrastructure protection: Requirements and challenges for the 21st century. In *International Journal of Critical Infrastructure Protection* . 2015. Vol. 8, p. 53–66. .

4. (PDF) Cyber Security of Critical Infrastructures. In *ResearchGate* [interaktyvus]. [žiūrėta 2019-10-30]. Prieiga per internetą: <https://www.researchgate.net/publication/322909940_Cyber_Security_of_Critical_Infrastructures>.

5. WOOD, A. ir kt. A security architectural pattern for risk management of industry control systems within critical national infrastructure. In *IJCIS* . 2017. Vol. 13, p. 113–132. .

6. Risks to Critical Infrastructure That Use Cloud Services. In . p. 1. .

7. The Twelve-Factor App. In [interaktyvus]. [žiūrėta 2019-11-02]. Prieiga per internetą: <https://12factor.net/>.

8. 2019 National Intelligence Strategy. In [interaktyvus]. [žiūrėta 2019-11-02]. Prieiga per internetą: <https://www.dni.gov/index.php/newsroom/reports-publications/item/1943-2019-national-intelligence-strategy>.

9. Cyber attacks targeting critical infrastructure | IEC e-tech | Issue' 02/2019. In *IEC e-tech* [interaktyvus]. [žiūrėta 2019-10-26]. Prieiga per internetą: <https://www.iecetech.org/index.php/Technology-Focus/2019-02/Cyber-attacks-targeting-critical-infrastructure>.

10. [Interaktyvus]. 2019. [žiūrėta 2019-11-02]. Prieiga per internetą: <https://ics-cert.kaspersky.com/reports/2019/03/27/threat-landscape-for-industrial-automation-systems-h2-2018/>.

11. ICS-CERT Landing | CISA. In [interaktyvus]. [žiūrėta 2019-11-02]. Prieiga per internetą: <https://www.us-cert.gov/ics>.

12. OGIE, R.I. Cyber Security Incidents on Critical Infrastructure and Industrial Networks. In *Proceedings of the 9th International Conference on Computer and Automation Engineering - ICCAE '17* [interaktyvus]. Sydney, Australia: ACM Press, 2017. p. 254–258. [žiūrėta 2019-10-27]. Prieiga per internetą: <http://dl.acm.org/citation.cfm?doid=3057039.3057076>.

13. NAGARAJU, S. - PARTHIBAN, L. SecAuthn: Provably Secure Multi-Factor Authentication for the Cloud Computing Systems. In . 2016. .

14. What is Spear Phishing? | Definition and Risks | Kaspersky. In [interaktyvus]. [žiūrėta 2019-11-10]. Prieiga per internetą: <https://www.kaspersky.com/resource-center/definitions/spear-phishing>.

15. NIST Special Publication 800-63-3. In [interaktyvus]. [žiūrėta 2019-10-26]. Prieiga per internetą: </sp800-63-3.html>.

16. MICHAEL SCOVETTA [interaktyvus]. .22:57:26 UTC. [žiūrėta 2019-11-03]. Prieiga per internetą: <https://www.slideshare.net/scovetta/wp-consumer-passwordworstpractices-10597735>.

17. OMETOV, A. ir kt. Multi-Factor Authentication: A Survey. In *Cryptography* . 2018. Vol. 2, no. 1, p. 1. .

18. What are the different ways to implement Multifactor Authentication? In *Auth0 - Blog* [interaktyvus]. [žiūrėta 2019-11-16]. Prieiga per internetą: <https://auth0.com/blog/different-ways-to-implement-multifactor/>.

19. SMS-based two-factor authentication is not safe — consider these alternative 2FA methods instead. In [interaktyvus]. [žiūrėta 2019-11-16]. Prieiga per internetą: <https://www.kaspersky.com/blog/2fa-practical-guide/24219/>.

20. NIST Special Publication 800-63B. In [interaktyvus]. [žiūrėta 2019-11-16]. Prieiga per internetą: </sp800-63b.html>.

21. KRAWCZYK, H. ir kt. HMAC: Keyed-Hashing for Message Authentication. In [interaktyvus]. [žiūrėta 2019-11-16]. Prieiga per internetą: <https://tools.ietf.org/html/rfc2104>.

22. RYDELL, J. ir kt. TOTP: Time-Based One-Time Password Algorithm. In [interaktyvus]. [žiūrėta 2019-10-26]. Prieiga per internetą: <https://tools.ietf.org/html/rfc6238>.

23. RSA SecurID Hardware Tokens | Store. In *RSA.com* [interaktyvus]. [žiūrėta 2019-11-30]. Prieiga per internetą: <https://www.rsa.com/en-us/products/rsa-securid-suite/rsa-securid-access/securid-hardware-tokens/rsa-securid-hardware-tokens>.

24. RSA SecurID Software Token for Microsoft Windows | RSA Link. In [interaktyvus]. [žiūrėta 2019-11-30]. Prieiga per internetą: <https://community.rsa.com/community/products/securid/software-token-windows>.

25. AUTHY Push Authentication: Bringing the Most Secure Method of 2FA Mainstream. In *Medium* [interaktyvus]. 2018. [žiūrėta 2019-11-30]. Prieiga per internetą: <https://medium.com/@Authy/push-authentication-bringing-the-most-secure-method-of-2fa-mainstream-d45aef238cc8>.

26. LYNES, M. ir kt. [interaktyvus]. 2016. [žiūrėta 2019-10-26]. Prieiga per internetą: <https://patents.google.com/patent/US9232339B2/en>.

27. Introduction to Push Notifications | Web. In *Google Developers* [interaktyvus]. [žiūrėta 2019-11-16]. Prieiga per internetą: <https://developers.google.com/web/ilt/pwa/introduction-to-push-notifications>.

28. ALDUMIJI, N.A. - KHAN, E.A. Fingerprint and location based multifactor authentication for mobile applications. In *International Journal of Engineering* . p. 13. .

29. ZHANG, F. ir kt. Location-Based Authentication and Authorization Using Smart Phones. In *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications* [interaktyvus]. Liverpool, United Kingdom: IEEE, 2012. p. 1285–1292. [žiūrėta 2019-11-17]. Prieiga per internetą: <http://ieeexplore.ieee.org/document/6296127/>.

30. SHAH, P. Image Based Authentication System. In . 2013. .

31. YubiKey-5-NFC-Free-Express-Shipping-from.jpg (400×400). In [interaktyvus]. [žiūrėta 2019-11-30]. Prieiga per internetą: <https://www.picclickimg.com/d/l400/pict/362779339376_/YubiKey-5-NFC-Free-Express-Shipping-from.jpg>.

32. Auth0 Guardian. In *Auth0* [interaktyvus]. [žiūrėta 2019-11-30]. Prieiga per internetą: <https://auth0.com/guardian>.

33. Adaptive Multi-Factor Authentication. In *Okta* [interaktyvus]. 2018. [žiūrėta 2019-11-30]. Prieiga per internetą: <https://www.okta.com/products/customer-identity/adaptive-multi-factor-authentication/>.

34. LTD, U.S. UNLOQ | Features And Secure Authentication Options. In [interaktyvus]. [žiūrėta 2019-11-30]. Prieiga per internetą: <https://unloq.io>.

35. Multi-Factor Authentication (MFA). In *Duo Security* [interaktyvus]. [žiūrėta 2019-11-30]. Prieiga per internetą: <https://duo.com/product/multi-factor-authentication-mfa>.

36. Identity and Access Management | RSA SecurID Suite. In *RSA.com* [interaktyvus]. [žiūrėta 2019-11-30]. Prieiga per internetą: <https://www.rsa.com/en-us/products/rsa-securid-suite>.

37. NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY [interaktyvus]. .Gaithersburg, MD: National Institute of Standards and Technology, 2018. [žiūrėta 2019-10-29]. Prieiga per internetą: <http://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf>.

38. JOINT TASK FORCE TRANSFORMATION INITIATIVE [interaktyvus]. .[s.l.]: National Institute of Standards and Technology, 2013. [žiūrėta 2019-11-10]. Prieiga per internetą: <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf>.

39. STOUFFER, K. ir kt. [interaktyvus]. .[s.l.]: National Institute of Standards and Technology, 2015. [žiūrėta 2019-11-16]. Prieiga per internetą: <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf>.

40. CIS Critical Control 12: Boundary Defense Explained. In *Rapid7 Blog* [interaktyvus]. 2018. [žiūrėta 2019-11-16]. Prieiga per internetą: <https://blog.rapid7.com/2018/04/02/cis-critical-control-12-boundary-defense-explained/>.

41. 20 CIS Controls: Control 12 - Boundary Defense. In *The State of Security* [interaktyvus]. 2018. [žiūrėta 2019-11-16]. Prieiga per internetą: <https://www.tripwire.com/state-of-security/security-data-protection/20-critical-security-controls-control-12-boundary-defense/>.

42. The 20 CIS Controls & Resources. In *CIS* [interaktyvus]. [žiūrėta 2019-11-16]. Prieiga per internetą: <https://www.cisecurity.org/controls/cis-controls-list/>.

43. Seven Steps to Effectively Defend Industrial Control Systems_S508C.pdf. In [interaktyvus]. [žiūrėta 2019-11-16]. Prieiga per internetą: <https://www.us-cert.gov/sites/default/files/documents/Seven%20Steps%20to%20Effectively%20Defend%20Industrial%20Control%20Systems_S508C.pdf>.

44. Advanced Solutions for Critical Infrastructure Protection. In . p. 18. .

45. Industrial Control Systems Remote Access Protocol | Cyber.gov.au. In [interaktyvus]. [žiūrėta 2019-11-19]. Prieiga per internetą: <https://www.cyber.gov.au/publications/industrial-control-systems-remote-access-protocol>.

46. Configuring and Managing Remote Access for Industrial Control Systems. In . p. 66. .

47. THE SMART GRID INTEROPERABILITY PANEL–SMART GRID CYBERSECURITY COMMITTEE [interaktyvus]. .[s.l.]: National Institute of Standards and Technology, 2014. [žiūrėta 2019-11-16]. Prieiga per internetą: <https://nvlpubs.nist.gov/nistpubs/ir/2014/NIST.IR.7628r1.pdf>.

48. ABDELLAOUI, A. ir kt. A Novel Strong Password Generator for Improving Cloud Authentication. In *Procedia Computer Science* . 2016. Vol. 85, p. 293–300. .

49. COPY, B. ir kt. C2MON SCADA deployment on CERN cloud infrastructure. In . 2018. p. THBPL01. .

50. (1) (PDF) Hosting critical infrastructure services in the cloud environment considerations. In *ResearchGate* [interaktyvus]. [žiūrėta 2019-11-16]. Prieiga per internetą: <https://www.researchgate.net/publication/268448399_Hosting_critical_infrastructure_services_in_the_cloud_environment_considerations>.

51. S107381-IoT-in-Oil-and-Gas-Architectures-and-Experiences.pdf. In [interaktyvus]. [žiūrėta 2019-11-24]. Prieiga per internetą: <https://docs.huihoo.com/redhat/summit/2017/S107381-IoT-in-Oil-and-Gas-Architectures-and-Experiences.pdf>.

52. cyber-pitbull-mls-containers-presentation.pdf. In [interaktyvus]. [žiūrėta 2019-11-24]. Prieiga per internetą: <https://gdmissionsystems.com/-/media/General-Dynamics/Cyber-and-Electronic-Warfare-Systems/PDF/Brochures/cyber-pitbull-mls-containers-presentation.ashx?la=en&hash=F8A86569590A2A8D7689A354E1637F2BC812CAF0>.

53. What is a Container? In *Docker* [interaktyvus]. [žiūrėta 2019-11-10]. Prieiga per internetą: <https://www.docker.com/resources/what-container>.

54. Production-Grade Container Orchestration. In [interaktyvus]. [žiūrėta 2019-11-10]. Prieiga per internetą: <https://kubernetes.io/>.

55. A Beginner's Guide to Kubernetes - ContainerMind - Medium. In [interaktyvus]. [žiūrėta 2019-11-09]. Prieiga per internetą: <https://medium.com/containermind/a-beginners-guide-to-kubernetes-7e8ca56420b6>.

56. kubernetes/dashboard: General-purpose web UI for Kubernetes clusters. In [interaktyvus]. [žiūrėta 2019-11-09]. Prieiga per internetą: <https://github.com/kubernetes/dashboard>.

57. The OAuth 2.0 Authorization Framework: Bearer Token Usage. In [interaktyvus]. [žiūrėta 2019-11-10]. Prieiga per internetą: <http://self-issued.info/docs/draft-ietf-oauth-v2-bearer.html>.

58. OAuth 2.0 — OAuth. In [interaktyvus]. [žiūrėta 2020-03-19]. Prieiga per internetą: <https://oauth.net/2/>.

59. AGARWAL, N. ir kt. Proof Key for Code Exchange by OAuth Public Clients. In [interaktyvus]. [žiūrėta 2020-03-19]. Prieiga per internetą: <https://tools.ietf.org/html/rfc7636>.

60. Firebase. In [interaktyvus]. [žiūrėta 2020-03-19]. Prieiga per internetą: <https://firebase.google.com/>.

61. Cross Site Scripting (XSS) | OWASP. In [interaktyvus]. [žiūrėta 2020-03-19]. Prieiga per internetą: <https://owasp.org/www-community/attacks/xss/>.

62. Cross Site Request Forgery (CSRF) | OWASP. In [interaktyvus]. [žiūrėta 2020-03-19]. Prieiga per internetą: <https://owasp.org/www-community/attacks/csrf>.

63. Hashing in Action: Understanding bcrypt. In *Auth0 - Blog* [interaktyvus]. [žiūrėta 2020-03-19]. Prieiga per internetą: <https://auth0.com/blog/hashing-in-action-understanding-bcrypt/>.

64. Ant Design - A UI Design Language and React UI library. In [interaktyvus]. [žiūrėta 2020-03-19]. Prieiga per internetą: <https://ant.design/>.

65. BONNEAU, J. ir kt. The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. In *2012 IEEE Symposium on Security and Privacy* [interaktyvus]. San Francisco, CA, USA: IEEE, 2012. p. 553–567. [žiūrėta 2019-10-27]. Prieiga per internetą: <http://ieeexplore.ieee.org/document/6234436/>.

66. LAMI, I.A. ir kt. LocBiometrics: Mobile phone based multi- factor biometric authentication with time and location assurance. In . p. 4. .

67. Internet Banking Login with Multi-Factor Authentication. In *KSII Transactions on Internet and Information Systems* [interaktyvus]. 2016. Vol. 11, no. 1. [žiūrėta 2021-02-11]. . Prieiga per internetą: <http://www.itiis.org/digital-library/manuscript/1586>.

68. MACIEJ, B. ir kt. Multifactor Authentication Protocol in a Mobile Environment. In *IEEE Access* . 2019. Vol. 7, p. 157185–157199. .

69. FANG, X. - ZHAN, J. Online Banking Authentication Using Mobile Phones. In *2010 5th International Conference on Future Information Technology* [interaktyvus]. Busan, Korea (South): IEEE, 2010. p. 1–5. [žiūrėta 2021-02-13]. Prieiga per internetą: <http://ieeexplore.ieee.org/document/5482634/>.

70. MISBAHUDDIN, M. ir kt. A Unique-ID based Usable Multi-Factor Authentication Scheme for e-Services. In . p. 7. .

71. HUSSEIN, K.W. Design and Implementation of Multi Factor Mechanism for Secure Authentication System. In . 2013. Vol. 11, no. 7, p. 7. .

72. REESE, K. ir kt. A Usability Study of Five Two-Factor Authentication Methods. In . p. 15. .

Priedai

1 priedas. Kritinės infrastruktūros grėsmių šaltiniai

Nr.	Grėsmės šaltinis	Galimybės	Pagrindimas	Motyvacija	Pagrindimas
TS1	Neapsikentę ar nesąžiningi darbuotojai	Aukštos	Žinios apie vidines sistemas ir procesus	Aukšta	Šio tipo grėsmės šaltinis turi aukštą tikimybę turėti siekį sukelti tam tikros formos žalą organizacijai
TS2	Užsienio žvalgybos tarnybos	Aukštos	Aukštos kvalifikacijos ir finansuojamos	Žema	*SCADA* sistemos dažniausiai neturi jautrios informacijos, kurios siektų užsienio žvalgybos tarnybos
TS3	Valstybių finansuojami	Aukštos	Aukštos kvalifikacijos ir finansuojamos	Aukšta	Tai jau buvo istoriškai įvykdyta (*Stuxnet* pavyzdys)
TS4	Įsilaužėliai mėgėjai	Žemos	Žema kvalifikacija/sugebėjimai	Vidutinė - aukšta	Dažniausias atakos vektorius yra neteisingai sukonfigūruotos viešai pasiekiamos sistemos. Įsilaužėliai mėgėjai linkę vykdyti pažeidžiamumų paiešką ir naudoti įrankius, nereikalaujančius gilių žinių
TS5	Žalingos programinės įrangos kūrėjai	Aukštos	Aukštos kvalifikacijos programuotojai	Žema	Didžioji dalis žalingos programinės įrangos nėra nukreipta į *SCADA/ICS* platformas
TS6	Teroristai	Vidutinės	Teroristai per kelis metus tapo įgudę kibernetinio karo srityje	Aukšta	Kritinė nacionalinė infrastruktūra yra perspektyvus atakos vektorius, kurį norėtų paveikti teroristai
TS7	Tiriantieji žurnalistai	Žemos	Menkų sugebėjimų, dažniausiai naudojasi išoriniais šaltiniais atakų vykdymui	Žema	*SCADA/ICS* sistemose nėra saugoma šiam grėsmės šaltiniui naudinga informacija
TS8	Komerciniai konkurentai (pramoninis šnipinėjimas)	Vidutinės	Įgūdžiai ir žinios apie sistemas ir procesus	Žema - vidutinė	Konkurencingos rinkos kainos ir išlaidos gali lemti konkurentų pramoninį šnipinėjimą
TS9	Politiniai aktyvistai	Žemos	Žema kvalifikacija ir tinklalapių gadinimas	Žema	Labiau nukreipta į tinklalapių atakas. Mažiau nukreipta į politines priežastis
TS10	Organizuotos kriminalinės grupuotės	Žemos - vidutinės	Žemi – vidutiniai sugebėjimai vykdant atakas prieš *SCADA/ICS* sistemas	Žema	Labiau nukreipta į kreditinių kortelių apgaules, pinigų plovimą ir kita

Nr.	Grėsmės šaltinis	Galimybės	Pagrindimas	Motyvacija	Pagrindimas
TS11	Mokslinė bendruomenė ir tyrimų grupės	Aukštos	Aukštos kvalifikacijos. Vykdantys naujausius tyrimus	Vidutinė	Mokslinė bendruomenė ir tyrimų grupės praneša apie pažeidžiamumus publikacijose ir seminaruose

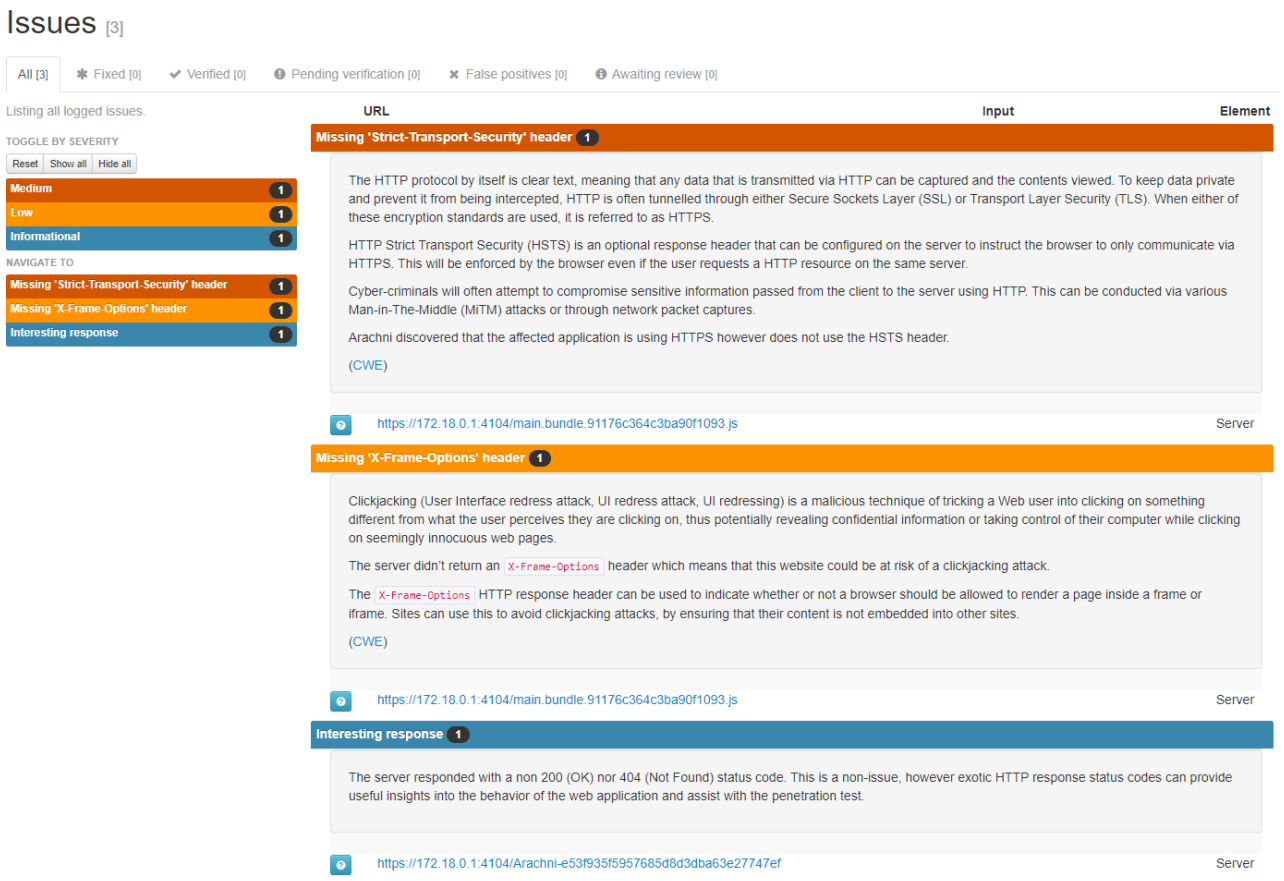
2 priedas. Rizikų įvertinimo ir šalinimo planas

Nr.	Rizika	Grėsmės šaltiniai	Poveikis	Sunkumas	Pašalinimas (kontrolė)	Likutinė rizika
R1	Neautorizuota prieiga prie valdymo sistemų per viešus tinklus	TS2, TS3, TS4, TS5, TS6, TS8	Aukštas	Aukštas	Perimetro apsauga (ugniasienė), įsibrovimo aptikimo ir prevencijos sistemos (*IDS/IPS*), proaktyvus paslaugų stebėjimas, auditas	Nėra
R3	Nepakankamas tinklo stebėjimas /auditas	TS1, TS2, TS3, TS4, TS5, TS6, TS8	Aukštas	Aukštas	Proaktyvus paslaugų stebėjimas, auditas	Nėra
R5	Nepakankamas suvokimas apie *ICS/SCADA* sistemų saugą	TS1, TS2, TS3, TS4, TS6, TS8	Vidutinis	Vidutinis	Saugos mokymai ir personalo sąmoningumo ugdymo programos diegimas, saugos politika	Nėra
R8	Duomenų ir komandų siuntimas iš dispečerio į valdymo tinklą atviru tekstu	TS2, TS3, TS6	Vidutinis	Vidutinis	Virtualaus *LAN* (*VLAN*) segregacija, komutatorių stiprinimas siekiant stebėti ir apsisaugoti nuo *CAM* užtvindymo atakų ir *VLAN* šuolių	Duomenys vis tiek išliks nešifruoti, jei nebus prieinamas saugus komunikavimo protokolas
R13	Nuotolinė priežiūra ir trečių šalių prieiga per nekontroliuojamus vartus (angl. *gateway*)	TS2, TS3, TS4, TS5, TS6, TS8	Aukštas	Aukštas	Sluoksniuota architektūra, tinklo įsibrovimo aptikimo ir prevencijos sistemos (*NIDS/NIPS*), proaktyvus stebėjimas, rolėmis grįstos prieigos teisės, trečių šalių saugos politika	Nėra

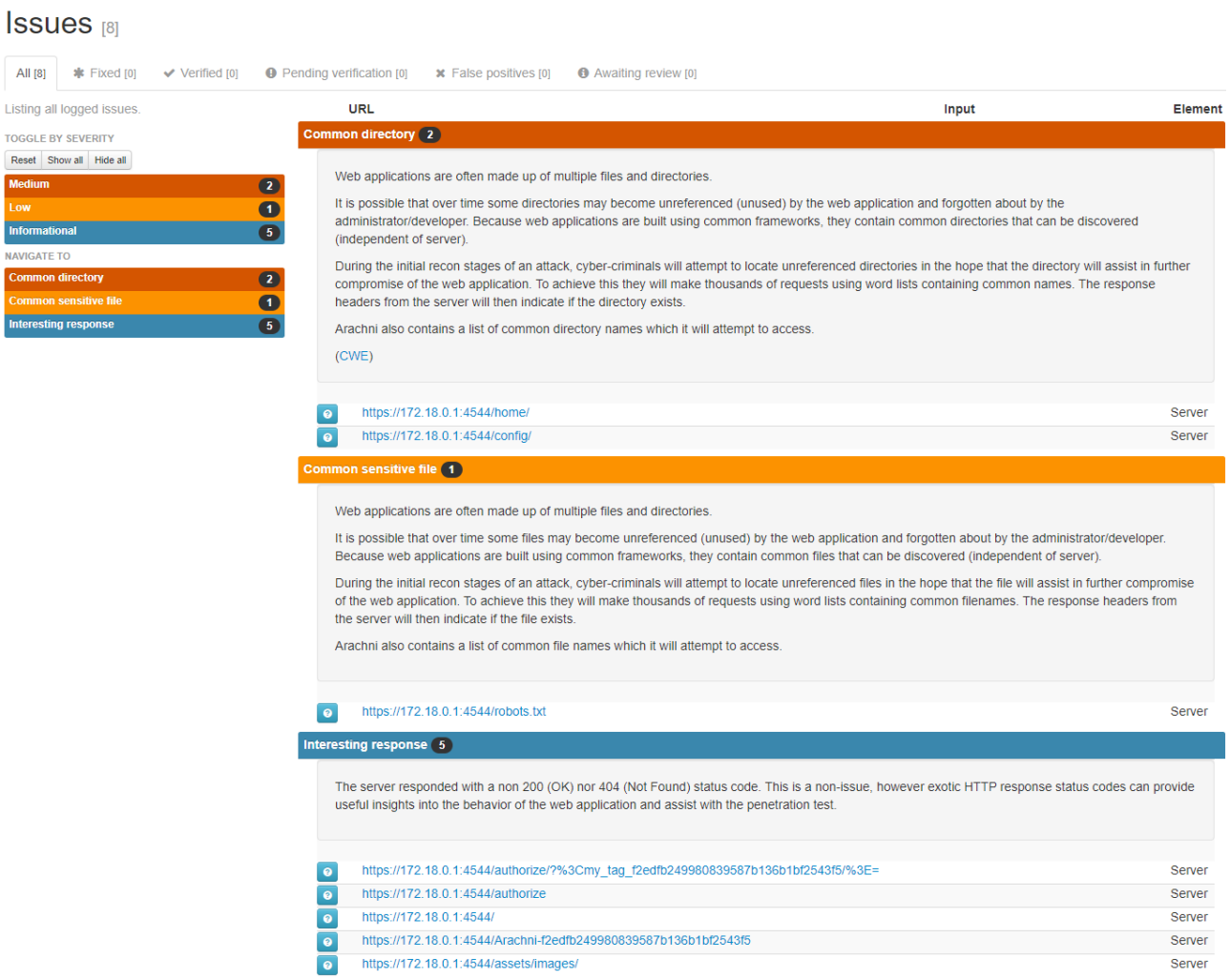
3 priedas. Komerciniai reikalavimai, skirti kritinės infrastruktūros sistemoms

Nr.	Reikalavimas
BR1	Paslauga turi užtikrinti maksimalų veikimo laiką ir užkirsti kelią kontrolės sistemų sutrikimams.
BR2	Paslauga turi būti palaikoma nuotoliniu būdu.
BR4	Paslauga gali būti pasiekiama tik autorizuoto personalo.
BR6	Paslauga turi vykdyti vidinį auditą ir stebėjimą, siekiant nustatyti neteisingą sistemos naudojimą ar ataką.

4 priedas. Dinaminės *WEB* serverių analizės rezultatai



81 pav. *UI/Gateway* serverio analizės rezultatai naudojant *Arachni* įrankį



82 pav. *Proxy* serverio analizės rezultatai naudojant *Arachni* įrankį

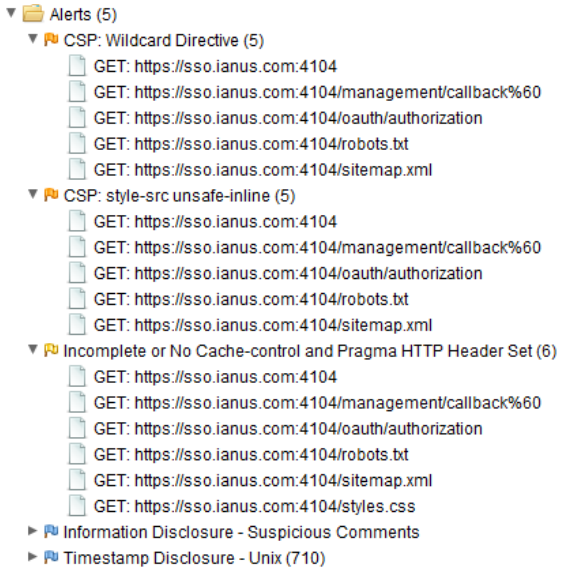
25 lentelė. *UI/Gateway* serverio analizės rezultatai naudojant *Nikto* įrankį

```
nikto -host sso.ianus.com -port 4104 -ssl -Plugins "@@DEFAULT;-sitefiles"

- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          10.0.2.2
+ Target Hostname:    sso.ianus.com
+ Target Port:        4104
---------------------------------------------------------------------------
+ SSL Info:        Subject:  /C=LT/ST=Lithuania/L=Kaunas/O=Ianus/CN=sso.ianus.com
                   Ciphers:  ECDHE-RSA-AES256-GCM-SHA384
                   Issuer:   /C=LT/ST=Lithuania/L=Kaunas/O=Ianus/CN=ianus
+ Start Time:         2020-12-05 18:03:33 (GMT2)
---------------------------------------------------------------------------
+ Server: nginx
+ The site uses SSL and Expect-CT header is not present.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ The Content-Encoding header is set to "deflate" this may mean that the server is vulnerable to the BREACH attack.
+ 7582 requests: 0 error(s) and 2 item(s) reported on remote host
```

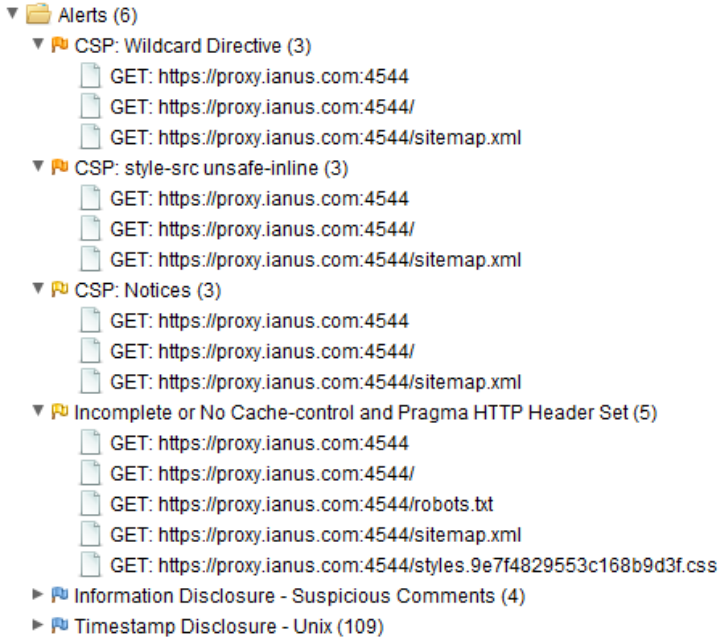
```
+ End Time:          2020-12-05 18:06:25 (GMT2) (172 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

26 lentelė. *Proxy* serverio analizės rezultatai naudojant *Nikto* įrankį

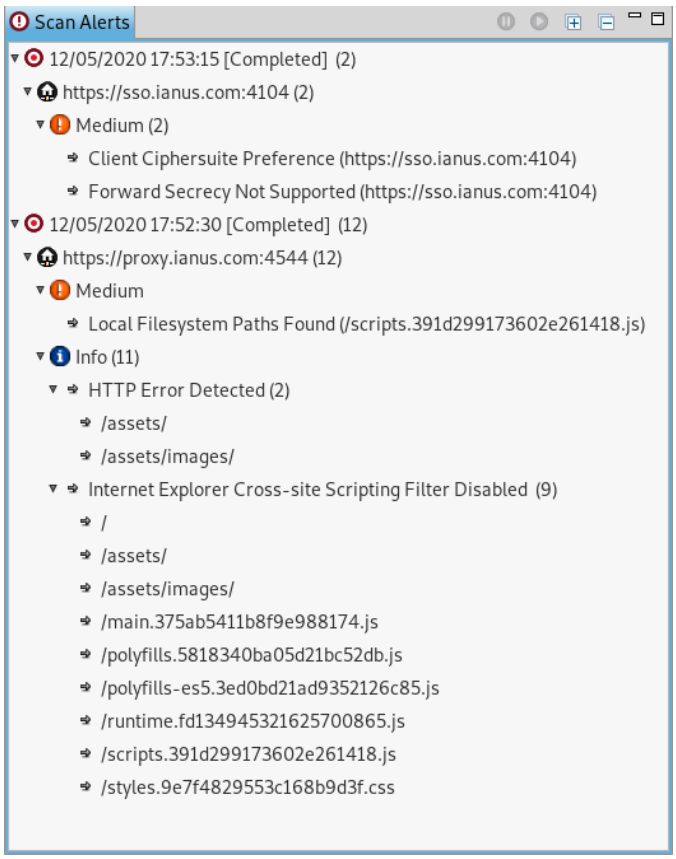
```
nikto -host proxy.ianus.com -port 4544 -ssl -Plugins "@@DEFAULT;-sitefiles"

- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          10.0.2.2
+ Target Hostname:    proxy.ianus.com
+ Target Port:        4544
---------------------------------------------------------------------------
+ SSL Info:        Subject:  /C=LT/ST=Lithuania/L=Kaunas/O=Ianus/CN=proxy.ianus.com
                   Ciphers:  TLS_AES_256_GCM_SHA384
                   Issuer:   /C=LT/ST=Lithuania/L=Kaunas/O=Ianus/CN=ianus
+ Start Time:         2020-12-05 18:13:20 (GMT2)
---------------------------------------------------------------------------
+ Server: nginx/1.17.10
+ X-XSS-Protection header has been set to disable XSS Protection. There is unlikely to be a good reason for this.
+ Uncommon header 'x-dns-prefetch-control' found, with contents: off
+ Expect-CT is not enforced, upon receiving an invalid Certificate Transparency Log, the connection will not be dropped.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ The Content-Encoding header is set to "deflate" this may mean that the server is vulnerable to the BREACH attack.
+ Retrieved x-powered-by header: Express
+ /config/: Configuration information may be available remotely.
+ OSVDB-3092: /home/: This might be interesting...
+ 7583 requests: 0 error(s) and 7 item(s) reported on remote host
+ End Time:          2020-12-05 18:16:32 (GMT2) (192 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```



83 pav. *UI/Gateway* serverio analizės rezultatai naudojant *ZAP* įrankį



84 pav. *Proxy* serverio analizės rezultatai naudojant *ZAP* įrankį

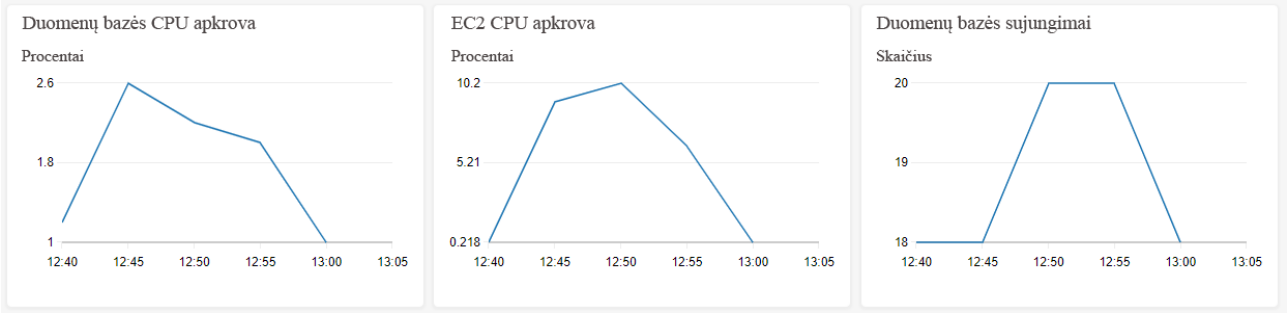


85 pav. Įrankio *Vega* aptiktų pažeidžiamumų sąrašas

5 priedas. Autentifikacijos *API* prieigos taškų greitaveikos tyrimo rezultatai

27 lentelė. Eksperimento naudojant duomenų rinkinį #2 greitaveikos rezultatai

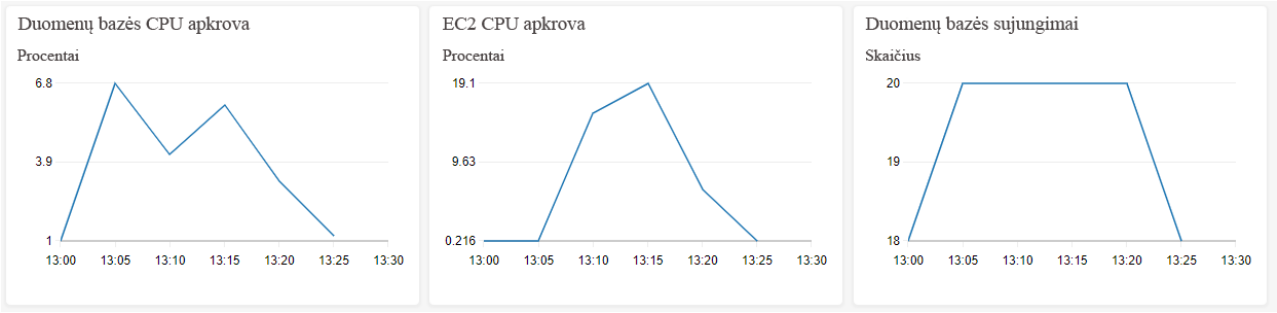
Prieigos taško Nr.	Imtis	Vidurkis (ms)	Minimumas (ms)	Maksimumas (ms)	Mediana (ms)	90 procentilis (ms)	95 procentilis (ms)
1.	780	112.43	102	730	107	128	142
2.	780	9.91	4	476	7	17	25
3.	780	15.66	7	111	11	23	48
4.	780	7.97	4	61	6	13	17.95
5.	780	14.2	7	110	11	20	32
6.	780	8.81	6	51	8	12	14
7.	774	16.18	8	152	12	22	37.5
8.	760	7.52	4	62	6	10	12



86 pav. Infrastruktūros metrikos, vykdant eksperimentą su duomenų rinkiniu #2

28 lentelė. Eksperimento naudojant duomenų rinkinį #3 greitaveikos rezultatai

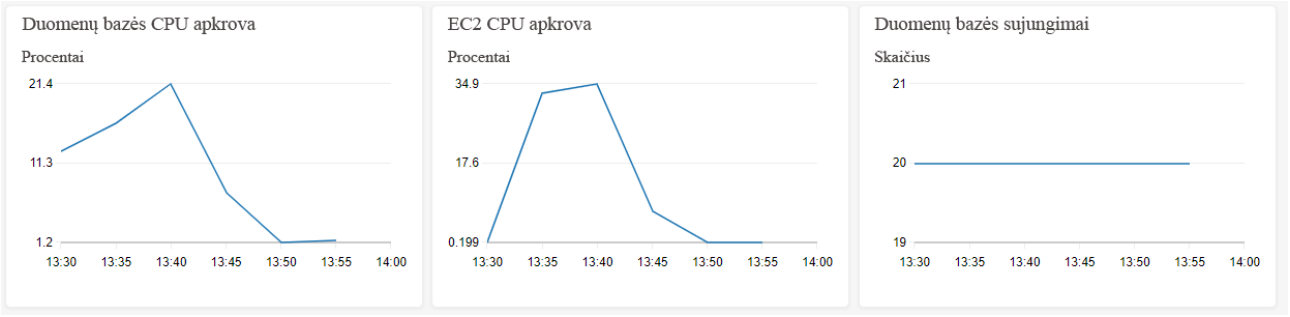
Prieigos taško Nr.	Imtis	Vidurkis (ms)	Minimumas (ms)	Maksimumas (ms)	Mediana (ms)	90 procentilis (ms)	95 procentilis (ms)
1.	1950	149.8	101	2762	108	233	271.45
2.	1950	9.82	3	494	7	18	25
3.	1950	25.5	6	2419	9	47	88
4.	1950	7.84	3	154	5	15	23
5.	1936	21.74	6	2033	9	34	71
6.	1902	35.44	19	161	29	59	71.85
7.	1900	24.79	7	2347	10	44	81
8.	1900	14.6	4	2131	6	18	37



87 pav. Infrastruktūros metrikos, vykdant eksperimentą su duomenų rinkiniu #3

29 lentelė. Eksperimento naudojant duomenų rinkinį #4 greitaveikos rezultatai

Prieigos taško Nr.	**Imtis**	**Vidurkis (ms)**	**Minimumas (ms)**	**Maksimumas (ms)**	**Mediana (ms)**	**90 procentilis (ms)**	**95 procentilis (ms)**
1.	3900	169.85	101	645	131	300	359
2.	3899	10.48	3	791	6	17	25
3.	3894	29.23	6	265	10	90	113
4.	3852	7.73	3	135	5	14	20
5.	3801	24.71	6	275	10	66	93
6.	3800	72.43	30	455	59	131	155
7.	3800	27.85	7	312	11	79	98.95
8.	3800	16.56	4	272	6	44	56

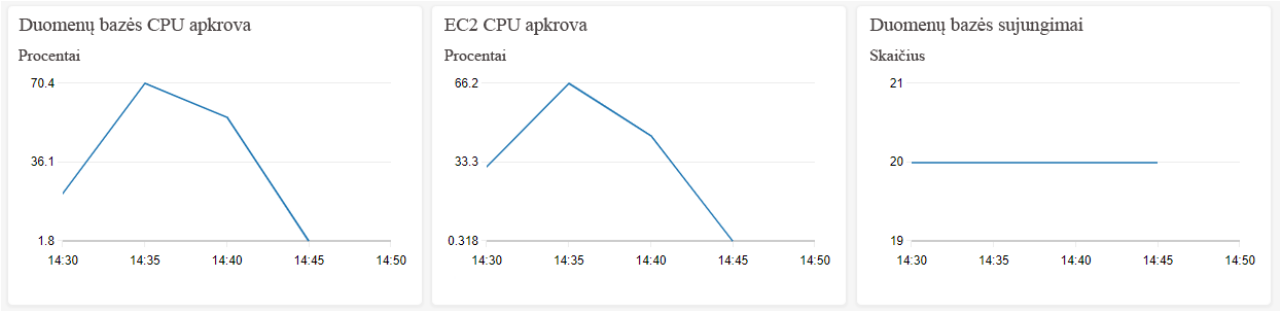


88 pav. Infrastruktūros metrikos, vykdant eksperimentą su duomenų rinkiniu #4

30 lentelė. Eksperimento naudojant duomenų rinkinį #5 greitaveikos rezultatai

Prieigos taško Nr.	**Imtis**	**Vidurkis (ms)**	**Minimumas (ms)**	**Maksimumas (ms)**	**Mediana (ms)**	**90 procentilis (ms)**	**95 procentilis (ms)**
1.	7600	237.1	101	1263	212	378	447
2.	7600	8.5	3	461	8	13	14
3.	7581	87.87	6	827	61	194	260
4.	7467	7.09	2	51	7	12	13

Prieigos taško Nr.	Imtis	Vidurkis (ms)	Minimumas (ms)	Maksimumas (ms)	Mediana (ms)	90 procentilis (ms)	95 procentilis (ms)
5.	7401	87.35	6	750	64	197	251
6.	7400	245.65	56	1276	210	462.9	570
7.	7400	79.98	7	1009	58	172	229
8.	7400	58.19	4	840	31	154	209



89 pav. Infrastruktūros metrikos, vykdant eksperimentą su duomenų rinkiniu #5

6 priedas. Įsiskverbimo testavimo scenarijų vykdymo iliustracijos

31 lentelė. Įsiskverbimo testų *HTTP* užklausos ir atsakymai

***RequestCreationBruteForcePassword* testas**	
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request* *Body:* *{* *"password": "password",* *"clientId": "security-test-client-3",* *"redirectUrl": "test-redirect-url",* *"codeChallenge": "ba7816bf8f01cfea414140de5dae2223 b00361a396177a9cb410ff61f20015ad",* *"username": "test-user"* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "This user is locked and cannot use any application. Please contact your administrator."* *}* *}*
***RequestCreationTamperRedirectUrl* testas**	
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request* *Body:*

	{ *"password": "password",* *"clientId": "security-test-client-3",* *"redirectUrl": "tampered-redirect-url", // Suklastota reikšmė* *"codeChallenge": "ba7816bf8f01cfea414140de5dae2223b00361a3 96177a9cb410ff61f20015ad",* *"username": "test-user"* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Provided redirect url does not match allowed redirect urls for this application."* *}* *}*
***RequestCreationTamperClientId* testas**	
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/token* *Body:* *{* *"clientId": "security-test-client-4", // Suklastota reikšmė* *"code": "433182cc-0c7f-4cf0-bdf9-2452a5d47d5e",* *"codeVerifier": "abc",* *"clientSecret": "security-test-client-secret",* *"grantType": "authorization_code"* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Oauth client was not found"* *}* *}*
***ApproveAuthenticationInvalidSignature* testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3576,* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHENTICATION",* *"iat": 1614929911,* *"sub": "3576",* *"exp": 1614930091*

	}
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/B9E9C87D/approve-authentication* *Body:* *{* *"token": " eyJraWQiOjM1NzYsImFsZyI6IlJTMjU2In0..."* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Failed to verify token"* *}* *}*
***ApproveAuthorizationInvalidSignature* testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3580,* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHORIZATION",* *"iat": 1614930118,* *"sub": "3580",* *"exp": 1614930298* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/14371A11/approve-authorization* *Body:* *{* *"token": "eyJraWQiOjM1ODAsImFsZyI6IlJTMjU2In0..."* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Failed to verify token"* *}* *}*
***ApproveAuthenticationNoneAlgorithm* testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3582,* *"alg": "none" // Suklastota reikšmė* *}* *{*

	"action": "APPROVE_AUTHENTICATION", *"iat": 1614947252,* *"sub": "3582",* *"exp": 1614947432* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/10A18FBD/approve-authentication* *Body:* *{* *"token": "eyJraWQiOjM1ODIsImFsZyI6Im5vbmUifQ..."* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Failed to verify token"* *}* *}*
***ApproveAuthorizationNoneAlgorithm* testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3586,* *"alg": "none" // Suklastota reikšmė* *}* *{* *"action": "APPROVE_AUTHORIZATION",* *"iat": 1614947437,* *"sub": "3586",* *"exp": 1614947617* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/18C53228/approve-authorization* *Body:* *{* *"token": "eyJraWQiOjM1ODYsImFsZyI6Im5vbmUifQ..."* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Failed to verify token"* *}* *}*
***ApproveAuthenticationTamperNotBefore* testas**	

Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3588,* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHENTICATION",* *"iat": 1614947565,* *"sub": "3588",* *"exp": 1614947745,* *"nbf": 1615068000 // Suklastota reikšmė* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/C1873851/approve-authentication* *Body:* *{* *"token": "eyJraWQiOjM1ODgsImFsZyI6IlJTMjU2In0... "* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Failed to verify token"* *}* *}*
***ApproveAuthorizationTamperNotBefore* testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3592,* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHORIZATION",* *"iat": 1614947731,* *"sub": "3592",* *"exp": 1614947911,* *"nbf": 1615068000 // Suklastota reikšmė* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/90558776/approve-authorization* *Body:* *{* *"token": "eyJraWQiOjM1OTIsImFsZyI6IlJTMjU2In0... "* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",*

	"data": { *"code": "error.validation",* *"message": "Failed to verify token"* *}* *}*
ApproveAuthorizationApproveOtherRequest **testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3599, // Suklastota reikšmė* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHORIZATION",* *"iat": 1614947912,* *"sub": "3599",* *"exp": 1614948092* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/B9B1F959/approve-authorization* *Body:* *{* *"token": "eyJraWQiOjM1OTksImFsZyI6IlJTMjU2In0... "* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Invalid attempt to approve the request"* *}* *}*
ApproveAuthenticationApproveOtherRequest **testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3631, // Suklastota reikšmė* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHENTICATION",* *"iat": 1614949467,* *"sub": "3631",* *"exp": 1614949647* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/7FF10BCA/approve-authentication* *Body:* *{* *"token": "eyJraWQiOjM2MzEsImFsZyI6IlJTMjU2In0... "* *}*

HTTP atsakymo paketas prieš pašalinant pažeidžiamumą	*HTTP/1.1 200* *Content-Type: application/json* *{* *"status": "success",* *"data": {* *"requestStatus": "PENDING_AUTHORIZATION_APPROVAL",* *"requestId": "7FF10BCA"* *}* *}*
HTTP atsakymo paketas pašalinus pažeidžiamumą	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Failed to verify token"* *}* *}*
***ApproveAuthenticationTamperExpiration* testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3603,* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHENTICATION",* *"iat": 1614948113,* *"sub": "3603",* *"exp": 1614722400 // Suklastota reikšmė* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/4FC0A88C/approve-authentication* *Body:* *{* *"token": "eyJraWQiOjM2MDMsImFsZyI6IlJTMjU2In0... "* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Failed to verify token"* *}* *}*
***ApproveAuthorizationTamperExpiration* testas**	
Dekoduotas užklausoje	*{* *"kid": 3607,* *"alg": "RS256"*

pateikiamas *JWT* žetonas	*}* *{* *"action": "APPROVE_AUTHORIZATION",* *"iat": 1614948222,* *"sub": "3607",* *"exp": 1614722400 // Suklastota reikšmė* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/6F0B7059/approve-authorization* *Body:* *{* *"token": "eyJraWQiOjM2MDcsImFsZyI6IlJTMjU2In0... "* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Failed to verify token"* *}* *}*
***ApproveAuthenticationTamperAction* testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3638, // Suklastota reikšmė* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHENTICATION", // Suklastota reikšmė* *"iat": 1614949639,* *"sub": "3638",* *"exp": 1614949819* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/9FEB3D88/approve-authentication* *Body:* *{* *"token": "eyJraWQiOjM2MzgsImFsZyI6IlJTMjU2In0... "* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Failed to verify token"* *}* *}*

***ApproveAuthorizationTamperAction* testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3639, // Suklastota reikšmė* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHORIZATION", // Suklastota reikšmė* *"iat": 1614949736,* *"sub": "3639",* *"exp": 1614949916* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/90E8ED03/approve-authorization* *Body:* *{* *"token": "eyJraWQiOjM2MzksImFsZyI6IlJTMjU2In0..."* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Invalid attempt to approve the request"* *}* *}*
***ApproveAuthenticationReplay* testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3612,* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHENTICATION",* *"iat": 1614948708,* *"sub": "3612",* *"exp": 1614948888* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/1241C3A4/approve-authentication* *Body:* *{* *"token": "eyJraWQiOjM2MTIsImFsZyI6IlJTMjU2In0... "* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {*

	"code": "error.validation", *"message": "The request cannot be approved"* *}* *}*
ApproveAuthorizationReplay **testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3616,* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHORIZATION",* *"iat": 1614948780,* *"sub": "3616",* *"exp": 1614948960* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/4F7D2F41/approve-authorization* *Body:* *{* *"token": "eyJraWQiOjM2MTYsImFsZyI6IlJTMjU2In0... "* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "The request cannot be approved"* *}* *}*
ApproveAuthenticationExpiredSession **testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3618,* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHENTICATION",* *"iat": 1614948853,* *"sub": "3618",* *"exp": 1614949033* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/3E778ECA/approve-authentication* *Body:* *{* *"token": "eyJraWQiOjM2MTgsImFsZyI6IlJTMjU2In0... "* *}*

HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Approval request has expired"* *}* *}*
***ApproveAuthorizationExpiredSession* testas**	
Dekoduotas užklausoje pateikiamas *JWT* žetonas	*{* *"kid": 3622,* *"alg": "RS256"* *}* *{* *"action": "APPROVE_AUTHORIZATION",* *"iat": 1614948933,* *"sub": "3622",* *"exp": 1614949113* *}*
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/request/72E87AC1/approve-authorization* *Body:* *{* *"token": "eyJraWQiOjM2MjIsImFsZyI6IlJTMjU2In0..."* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Approval request has expired"* *}* *}*
***ExchangeTokenExpiredRefreshToken* testas**	
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/token* *Body:* *{* *"clientId": "security-test-client-3",* *"clientSecret": "security-test-client-secret",* *"grantType": "refresh_token",* *"refreshToken": "bb4f451e-a576-4312-97e8-179821a2c2e1"* *// Suklastota reikšmė* *}*
HTTP atsakymo paketas	*HTTP/1.1 401* *Content-Type: application/json* *{*

	"status": "error", *"data": {* *"code": "error.unauthorized",* *"message": "Bad request"* *}* *}*
***ExchangeTokenTamperClientId* testas**	
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/token* *Body:* *{* *"clientId": "security-test-client-4", // Suklastota reikšmė* *"code": "a2c9959b-a741-4bb4-a907-55301909146e",* *"codeVerifier": "abc",* *"clientSecret": "security-test-client-secret",* *"grantType": "authorization_code"* *}*
HTTP atsakymo paketas prieš pašalinant pažeidžiamumą	*HTTP/1.1 200* *Content-Type: application/json* *{* *"status": "success",* *"data": {* *"accessToken": "eyJraWQiOiIxIiwiYWxnIjoiUlM1MTIifQ... ",* *"refreshToken": "17d0ab1c-a7c3-4309-a002-5abb3ebd0784"* *}* *}*
HTTP atsakymo paketas pašalinus pažeidžiamumą	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Oauth client was not found"* *}* *}*
***ExchangeTokenTamperClientSecret* testas**	
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/token* *Body:* *{* *"clientId": "security-test-client-3",* *"code": "8536a120-2ab5-4bab-af0c-fed009b1ba0e",* *"codeVerifier": "abc",* *"clientSecret": "invalid-secret", // Suklastota reikšmė* *"grantType": "authorization_code"* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json*

	{ *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Oauth client was not found"* *}* *}*
***ExchangeTokenTamperCodeVerifier* testas**	
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/token* *Body:* *{* *"clientId": "security-test-client-3",* *"code": "ee3fec73-bcb3-4dcf-a79e-8a4f8fdd53ff",* *"codeVerifier": "tampered-verifier", // Suklastota reikšmė* *"clientSecret": "security-test-client-secret",* *"grantType": "authorization_code"* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Invalid code verifier"* *}* *}*
***ExchangeTokenTamperReplay* testas**	
HTTP užklausos paketas	*Request method: POST* *Request URI: http://localhost:9002/api/oauth/token* *Body:* *{* *"clientId": "security-test-client-3",* *"code": "610aaddb-b049-4e4e-bfae-00b9eee8d255",* *"codeVerifier": "abc",* *"clientSecret": "security-test-client-secret",* *"grantType": "authorization_code"* *}*
HTTP atsakymo paketas	*HTTP/1.1 400* *Content-Type: application/json* *{* *"status": "error",* *"data": {* *"code": "error.validation",* *"message": "Authorization code was not found"* *}* *}*