VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ KATEDRA

Skaitmeninės tapatybės valdymas taikant blokų grandinę

Digital Identity Management using Blockchain

Bakalauro darbas

Atliko:	Jurgis Kargaudas	(parašas)
Darbo vadovas:	lekt. Aurimas Šimkus	(parašas)
Darbo recenzentas:	lekt. Andrius Adamonis	(parašas)

Vilnius – 2018

Santrauka

Bus pridėta parašius visą darbą.

Summary

To be added when the whole thesis is finished.

TURINYS

Įvadas

Interneto paslaugos šiais laikais yra neatsiejama žmonių gyvenimo dalis. Norėdami individualizuoti turinį, sustiprinti taikomosios programos saugumą ar siekdami iš anksto išvengti kenkėjiškų tikslų turinčių asmenų ar sukurtų robotų, paslaugų tiekėjai siekia identifikuoti savo naudotojus. Interneto naudotojų skaičiui perkopus 4 milijardus [**InternetUsers2018**], o kiekvienam naudotojui vidutiniškai turint po 7 skirtingas socialines paskyras [**Mander2017**], asmenų tapatybių valdymas, autentifikavimas ir autorizavimas tampa vis didesniu iššūkiu.

Tapatybių valdymas kelia problemų naudotojams. Bene didžiausi atsiradę keblumai: milžiniškas įsimintinų slaptažodžių kiekis bei sunkumai kontroliuojant savo asmens duomenų sklaidą skirtingose sistemose. Vidutiniškai interneto naudotojas turi 25 slaptažodžių reikalaujančias paskyras ir per dieną turi įvesti 8-is slaptažodžius [**Florencio2007**]. Susidarius tokiai situacijai, per didelis įsimintinų slaptažodžių kiekis neretai priverčia naudotojus paaukoti saugumą dėl patogumo ir pradėti naudoti tą patį slaptažodį sirtingoms sistemoms [**Pashalidis2003**; **Samar1999**]. Naudotojas, turėdamas keletą paskyrų skirtingose sistemose, taip pat praranda dalį savo asmens duomenų kontrolės. Jam tenka pasitikėti taikomosios programos naudojamomis technologijomis ir metodais ir tikėtis, kad jie bus pakankamai saugūs ir stabilūs bei suteikti asmens duomenys nepasieks nepageidaujamų adresatų. Didėjant naudojamų paslaugų kiekiui, naudotojo skaitmeninės tapatybės duomenis turi vis daugiau taikomųjų programų ir bent vienai iš jų patyrus programišių įsilaužimą ar kitokią nesėkmę, jautrūs naudotojo duomenys gali būti paviešinti.

Taikyti skirtingi metodai skaitmeninės tapatybės valdymo internete problemoms spręsti. Šiais laikais vienas dažniausiai internete sutinkamų sprendimų yra vienkartinis prisijungimas (angl. *Single Sign-On*). Šis sprendimas leidžia naudotojui pasirinkti vieną tapatybės tiekėją (angl. *identity provider*) ir patikėti jam skaitmeninės tapatybės valdymą. Tokiu būdu naudotojui pakanka turėti tik paskyrą tapatybės tiekėjo sistemoje bei kreipiantis į paslaugas prisijungti per jį. Tačiau šis sprendimo būdas taip pat turi aiškių trūkumų: naudotojas negali prisijungti prie paslaugų, nepalaikančių pasirinkto tapatybės tiekėjo, jo pasiekiamumas tampa vieninteliu nesėkmės tašku (angl. *single point of failure*), naudotojas taip pat praranda dalį savo asmens duomenų kontrolės. Naršantis internete asmuo yra priverstas pasitikėti tapatybės tiekėjo gebėjimu perduoti tik naudotojo leistus asmens duomenis ir tik toms trečiosioms šalims, kurias jis patvirtina. Kaip rodo *Cambridge Analytica* incidentas [**CambridgeAnalytica**], net didžiosios kompanijos, tokios kaip „Facebook“, ne visada sugeba tai užtikrinti.

Blokų grandinė (angl. *blockchain*) yra nauja alternatyva skaitmeninės tapatybės valdymui. Ši technologija veikia kaip paskirstytų įrašų platforma (angl. *distributed ledger platform*), kurioje kiekvienas įrašas yra nekintamas, o visi užfiksuoti įrašai atspindi tikslią transakcijų istoriją nuo pat grandinės sukūrimo [**Baars2016**]. Saugant tapatybės duomenis šioje grandinėje ir pritaikius reikiamą blokų grandinės pasiekiamumo lygį įrašų rašymui ir skaitymui, asmuo visada žinotų, kokia trečioji šalis gali pasiekti kokius tapatybės duomenis. Kadangi blokų grandinė yra decentralizuota, pritaikius ją skaitmeninių tapatybių valdyme taip pat būtų galima išvengti šioje srityje dažnos vienintelio nesėkmės taško problemos.

Šiame darbe blokų grandinės tinkamumas skaitmeninės tapatybės valdymui nagrinėja-

mas iš naudotojo perspektyvos. Pateikus esminius naudotojų poreikius identiteto valdymui, apžvelgiamas dabar naudojamų sistemų gebėjimas įgyvendinti šiuos reikalavimus. Įvertinus pagrindines neišspręstas naudotojams kylančias problemas, tiriama, kaip blokų grandinė gali padėti jas išspręsti, kokie tokio blokų grandinės panaudojimo skaitmeniniame tapatybės valdyme pranašumai, trūkumai bei pritaikymo barjerai (angl. *adoption barriers*).

Darbo tikslas - išanalizuoti blokų grandinės tinkamumą skaitmeninės tapatybės valdymui.

Darbe keliami uždaviniai:

1. Išskirti naudotojų poreikius skaitmeninės tapatybės valdymui internete.
2. Apžvelgti skaitmeninės tapatybės valdymo sprendimus, vertinant jų gebėjimą įgyvendinti išskirtas naudotojų reikmes.
3. Įvardyti naudotojams kylančias problemas skaitmeninės tapatybės valdyme.
4. Apibūdinti blokų grandinės technologiją ir jos savybes, leidžiančias spręsti įvardytas naudotojų problemas skaitmeninio identiteto srityje.
5. Pateikti blokų grandinės panaudojimo atvejį skaitmeninės tapatybės valdymui ir sukurti jo veikimą demonstruojantį prototipą.
6. Įvertinti pateiktą sprendimą apibūdinant jo privalumus, trūkumus ir galimus pritaikymo barjerus.

1. Skaitmeninės tapatybės valdymo apžvalga

Skaitmeniniame amžiuje tapatybės valdymas internete yra svarbi taikomųjų programų dalis. Šiame skyriuje apžvelgiamas asmenų identifikavimo internete poreikis, skaitmeninės tapatybės valdymo samprata, taikomi tapatybės valdymo modeliai bei pagrindinės naudotojams kylančios problemos. Skaitmeninės tapatybės, autentifikavimo, autorizavimo bei kitų darbe naudojamų terminų aiškinimai pateikiami skyriuje „Sąvokų apibrėžimai“.

1.1. Tapatybės atpažinimo poreikis

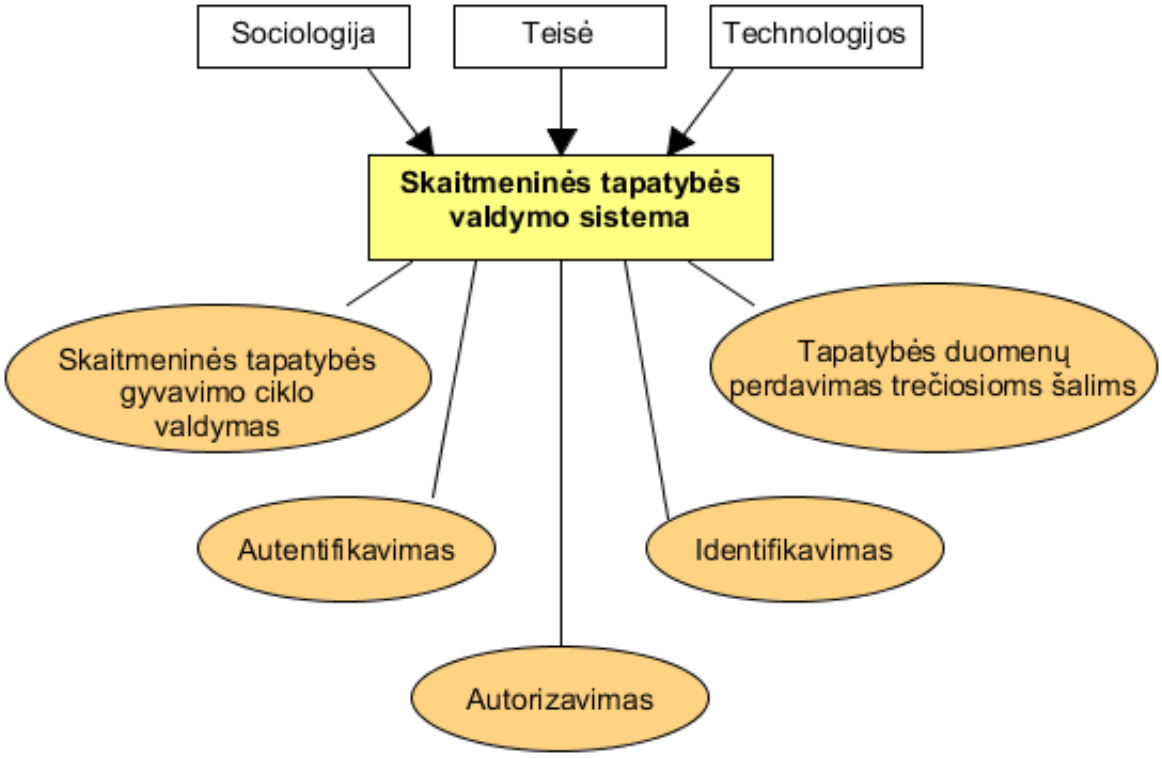
Naudotojo identifikavimas yra reikšminga interneto taikomųjų programų dalis. Paslaugų tiekėjai identifikuoja savo naudotojus norėdami [**RalucaBudiu2014**]:

- registruoti (angl. *log*) naudotojų veiklą,
- užtikrinti, kad naudotojas iš tikrųjų yra asmuo, kuris sakosi esąs,
- suteikti dalį funkcionalumo tik autorizuotiems naudotojams,
- individualizuoti tinklalapio ar taikomosios programos turinį pagal naudotojo poreikius,
- sukurti paslaugos naudotojų bendruomenę,
- išvengti galimų anoniminių naudotojų atakų.

Dėl išvardytų priežasčių naudotojų identifikavimas atlieka svarbią rolę įvairiose taikomųjų programų srityse - elektroninėje valdžioje, elektroninėje komercijoje, verslo sumanume (angl. *business intelligence*), tyrimuose bei saugume (angl. *homeland security*) [**Glasser2009**]. Kiekvienas paslaugų tiekėjas turi pasirinkti, kaip autentifikuoti, ir, jei reikia, autorizuoti naudotojus. Programos kūrėjas taip pat turi užtikrinti naudotojo suteiktų duomenų saugumą, o naudotojui tenka rūpintis skirtingų turimų paskyrų priežiūra ir savo duomenų sklaida tarp skirtingų sistemų. Minimus tapatybės atpažinimo skaitmeninėje erdvėje aspektus nagrinėja skaitmeninės tapatybės valdymo disciplina.

1.2. Skaitmeninės tapatybės valdymo samprata

Dėl nuolat vykstančios interneto ir jame esančių paslaugų plėtros tapatybių valdymo uždavinys pastaraisiais metais tapo itin svarbus [**Glasser2009**]. Skaitmeninės tapatybės valdymo pagrindinis uždavinys yra kontroliuoti tapatybę ir su ja susijusius procesus, tokius kaip autentifikavimas, autorizavimas, prieigų kontrolė, tapatybės gyvavimo ciklo valdymas bei saugus tapatybės atributų perdavimas trečiosioms šalims [**Cao2010**; **Dabrowski2008**]. Sprendžiant šį uždavinį, sukurta skirtingų skaitmeninės tapatybės valdymo sistemų. Šioms sistemoms įtaką daro kiti tapatybę nagrinėjantys mokslai (pvz. sociologija), taip pat jos gali atlikti keletą skirtingų funkcijų, susijusių su naudotojų tapatybe. Žemiau pateikiama diagrama, kurioje apibendrintas tapatybės valdymo sistemų kontekstas bei pagrindinės atliekamos užduotys:



1 pav. Skaitmeninių tapatybių valdymo sistemų kontekstas ir užduotys [**Glasser2009**]

Paveiksle matomos disciplinos turi skirtingą poveikį tapatybių valdymo sistemoms. Sociologija padeda apibrėžti tapatybę ir jos atitikmenį skaitmeninėje erdvėje, teisės mokslas nusako tapatybės duomenų naudojimo reikalavimus, o esamos technologijos formuoja sistemos įgyvendinimo niuansus. Verta pastebėti, kad tapatybės valdymo sistema gali atlikti ne visas diagramoje nurodomas funkcijas, o tik dalį iš jų.

1.3. Naudotojų poreikiai skaitmeninės tapatybės valdymo sistemoms

Skaitmeninės tapatybės valdymas yra plati sritis, kurią galima analizuoti iš skirtingų pusių: paslaugų tiekėjo, tapatybės tiekėjo ar naudotojo. Šiame darbe į skaitmeninių tapatybių valdymą žvelgta iš naudotojo perspektyvos - kaip skaitmeninio valdymo sistemos atitinka naudotojų poreikius bei lūkesčius. Išskirtos šios naudotojoms aktualios sistemų savybės:

- atpažinimo duomenų kiekis. Naudotojui vidutiniškai turint 25 paskyras, reikalaujančias slaptažodžių [**Florencio2007**] bei naudojant nuo 2 iki 12-os el. paštų [**Gross2007**], jis tampa priverstas prisiminti vis daugiau slaptažodžių bei identifikatorių. Atsimintinų autentifikavimo duomenų kiekiui augant, naudotojai yra linkę aukoti saugumą dėl patogumo ir naudoti panašius slaptažodžius skirtingose sistemose [**Pashalidis2003**; **Samar1999**];
- saugumas. Privatumas yra žmogaus poreikis ir visa visuomenė nukentėtų nuo jo nebuvimo [**Maliki2007**]. Suteikiant savo asmens duomenis internete naudotojai tikisi, kad jie bus patikimai saugomi ir nepasiekiami programišiams. Tapatybių valdymo sistemos turėtų būti

budrios saugumo rizikoms bei viešai skelbti saugumui skirtas priemones ir atliktų saugumo analizių rezultatus, kad tiek naudotojai, tiek paslaugų tiekėjai galėtų pasitikėti tapatybių valdymo sistemomis [**Dhamija2008**];

- asmens duomenų kontrolė. Pasak Nyderlanduose atliktų tyrimų, naudotojai nesijaučia kontroliuojantys savo asmens duomenų internete [**Baars2016**]. Dėl to naudotojai pradeda nepasitikėti taikomųjų programų kūrėjais, nes jie pilnai nežino, kokia informacija apie juos kaupiama ir kokioms sistemoms ji perduodama;
- patogumas (angl. *usability*). Naudotojams skaitmeninės tapatybės valdymas neretai yra tik pašalinis mechanizmas, reikalingas norint pasiekti paslaugą [**Dhamija2008**]. Dėl šios priežasties sistemos naudojimosi patogumas yra svarbus - kuo tapatybės valdymas yra labiau integruotas su asmens jau naudojamomis sistemomis, kuo mažiau jis reikalauja papildomo naudotojo įsitraukimo ir kuo suteikia geresnę naudotojo patirtį (angl. *user experience*), tuo labiau naudotojas bus linkęs pasirinkti šį identiteto valdymo sprendimą.

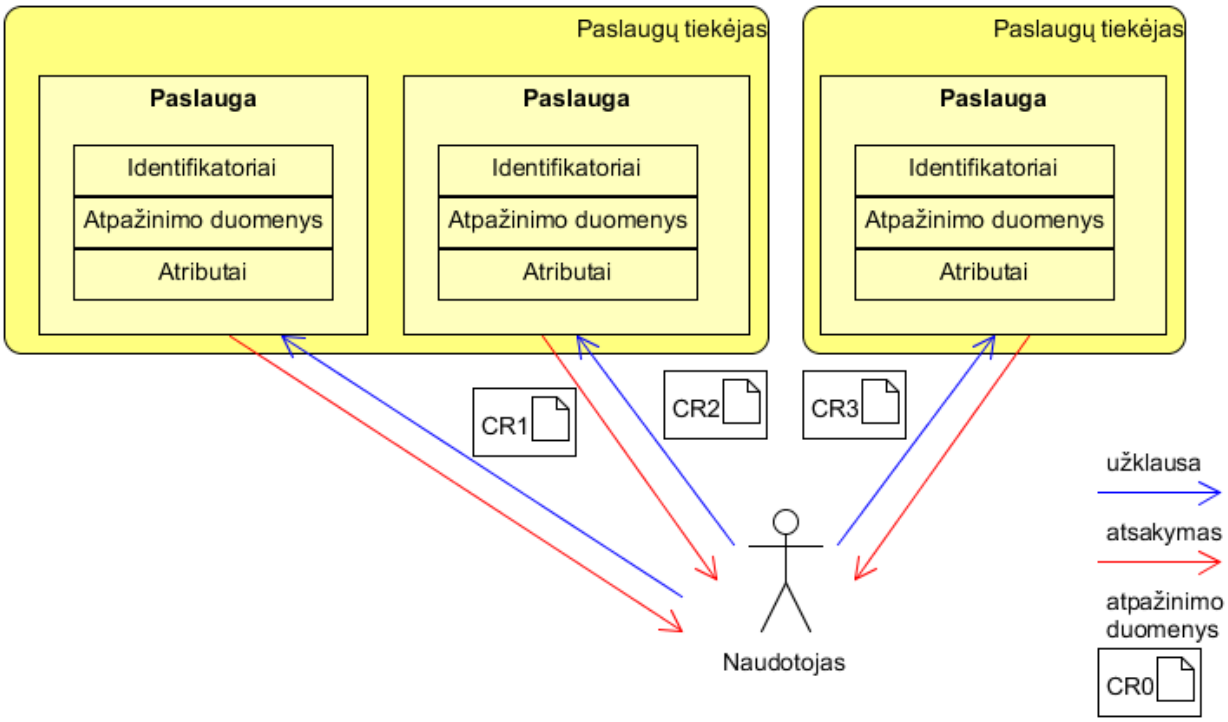
1.4. Skaitmeninės tapatybės valdymo modeliai

Naudojamų tapatybių valdymo sistemų architektūros bei veikimo principai yra skirtingi. S. Clauß ir M.Köhntopp savo tyrime pastebi, kad nėra vieningo standarto identiteto valdymo sistemoms [**Claus2001**]. Šiame skyriuje tiriami 3-ys dažniausiai naudojami identiteto valdymo modeliai. Tiriant kiekvieną modelį, pirmiausia apžvelgti jo bendri veikimo principai. Taip pat apžvelgtos paplitusios modelį įgalinančios technologijos (angl. *enabling technology*), nes modelio realizacijoje taikomi standartai ar protokolai gali turėti įtakos naudotojų poreikiams. Galiausiai, analizuotas modelio atitikimas naudotojų lūkesčiams, išvardytiems 1.3 skyrelyje.

1.4.1. Izoliuotas tapatybių valdymas

Modelis

Izoliuotame modelyje paslaugų tiekėjas yra ir tapatybės tiekėjas, nes visos su tapatybės valdymu susijusios operacijos yra atliekamos vieno serverio. Tapatybės duomenų saugojimas, autentifikavimas ir autorizavimas yra įgyvendinti paties paslaugų tiekėjo [**Cao2010**]. Kiekvienas naudotojas turi atskirus identifikatorius kiekvienai naudojamai paslaugai. Modelis grafiškai pavaizduotas žemiau esančiame paveiksle.



2 pav. Izoliuotas skaitmeninės tapatybės valdymas [**Cao2010**]

Pagal izoliuotą modelį, naudotojas turi savo paskyrą kiekvienoje naudojamoje sistemoje. Kiekvieną kartą autentifikuojant ar autorizuojant naudotoją, tai atlieka pats paslaugų tiekėjas, bendraudamas tiesiogiai su naudotoju (jo naršykle). Naudotojui prisijungus prie vieno tinklalapio ir gavus prieigos raktą, jis gali toliau naudotis šiuo tinklalapiu, tačiau prireikus pasinaudoti kita taikomąja programa, tapatybės atpažinimo veiksmai (autentifikavimas, autorizavimas) turi vėl būti atlikti naujoje sistemoje.

Realizacijos bei įgalinančios technologijos

Kadangi šis naudotojų autentifikavimo bei autorizavimo modelis naudojamas seniausiai, yra gana nemažai jį įgyvendinusių taikomųjų programų. Apsilankius keleto įmonių interneto tinklalapiuose, pastebėta, kad Lietuvoje naują paskyrą susikurti siūlo „Tiketa“, „Bilietai.lt“, „Pigu.lt“, „Varle.lt“, pasaulyje - „Booking.com“, „Skycop“, „AirBnB“, „CodinGame“ bei kitos platformos (išvardytų įmonių puslapiai tikrinti 2018 metų balandžio 14-ą dieną). Dalis iš jų remiasi ne vien tik savo izoliuotu tapatybės valdymu, bet jau turi į savo sistemas integravę ir papildomų autentifikavimo būdų (pvz. prisijungimą per „Facebook“ ar „Google“).

Realizacijų technologiniai sprendimai dažniausiai nėra viešai prieinami. Šiame modelyje kiekvienas paslaugų tiekėjas yra ir tapatybės tiekėjas, tad nereikia apibrėžti protokolų, duomenų formatų ar kitų detalių, kurios formalizuotų bendravimą tarp pasikliaujančiosios šalies ir tapatybės tiekėjo - visa tai pats nusprendžia ir įgyvendina paslaugų tiekėjas.

Naudotojų poreikių įgyvendinimas

Nors izoliuotas tapatybių valdymas yra gana paprastas paslaugų tiekėjams, tačiau jis greitai tampa nebekontroliuojamu naudotojams [**Josang2005**]. Jis verčia naudotojus turėti paskyrą kiekvienai paslaugai, o tai lemia daugybės identifikatorių ir slaptažodžių valdymą. Tai sukelia „slaptažodžių nuovargį“ (angl. *password fatigue*), o tai veda prie tų pačių identifikatorių ir slaptažodžių pasirinkimo skirtingoms paslaugoms [**Dhamija2008**].

Izoliuotame identiteto valdyme programišiams sunkiau atlikti sukčiavimo (angl. *phishing*) ataką, nes naudotojas nebūna nukreipiamas į tapatybės tiekėjo puslapį. Tai pagerina šio modelio saugumą. Dėl to, kad paslaugų tiekėjas yra ir tapatybės tiekėjas, šiame modelyje galima išvengti duomenų perdavimo tarp skirtingų serverių - tokiu būdu sumažėja ir rizika, kad šiuos duomenis jų persiuntimo metu perims programišius. Tačiau, standartų duomenų formatams bei perdavimui nebuvimas gali paskatinti paslaugų tiekėjus nepažvelgti į tai atsakingai ir įgyvendinti bendravimą su naudotojo naršykle atmestinai.

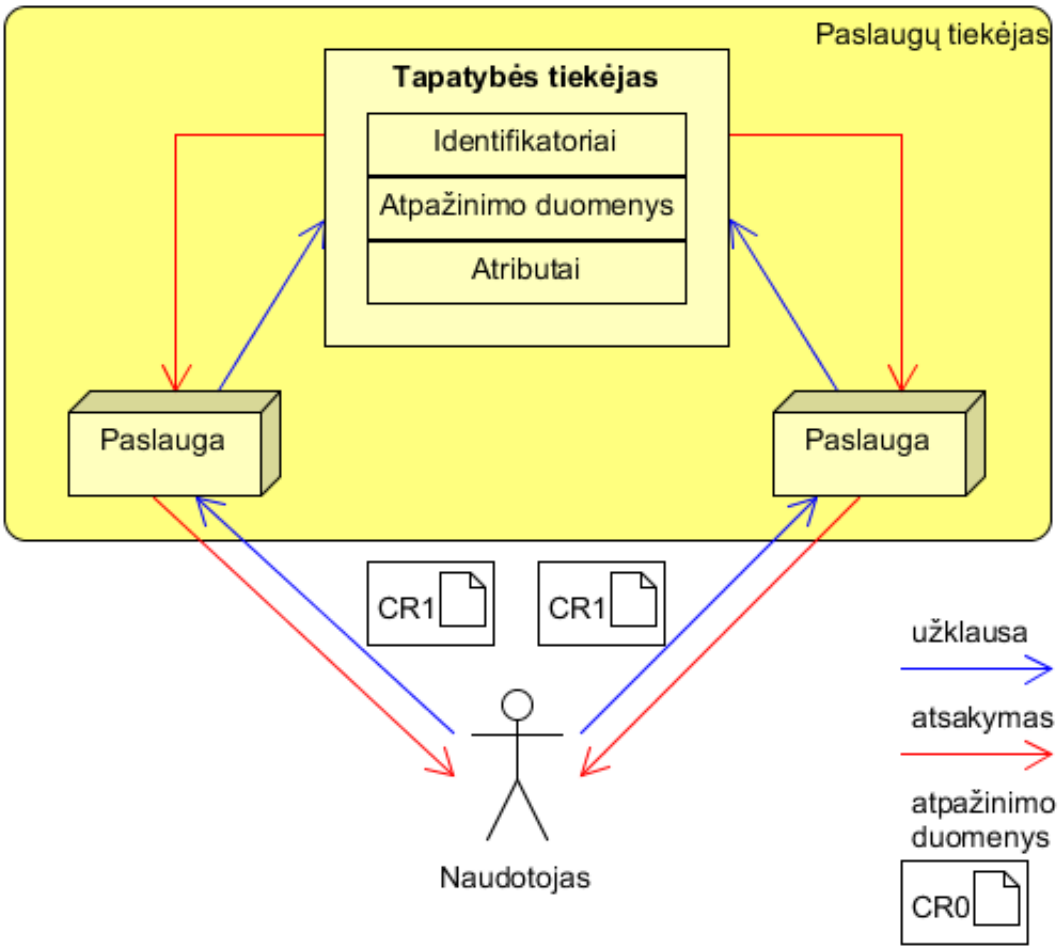
Naudotojų asmens duomenų kontrolė šiame modelyje priklauso nuo kiekvienos paslaugos. Asmenys atskirai suteikia savo duomenis kiekvienai paslaugai, dažniausiai paskyros sukūrimo metu. Jei paslauga informuoja apie duomenų panaudos atvejus (pvz. kam bus naudojamas el. pašto adresas), tuomet asmuo jausis labiau užtikrintas savo duomenų kontrole. Tačiau dėl šiame modelyje neišvengiamo duomenų suteikimo dideliam skirtingų paslaugų kiekiui, asmeniui tampa sunku prisiminti kiekvienos naudojamos platformos duomenų platinimo taisykles.

Izoliuotame tapatybės valdyme naudotojams tenka kartoti identifikavimo procesus (autentifikavimą, autorizavimą) tiek kartų, kiek paslaugų siekiama naudotis. Tai vargina naudotojus ir kuria blogą naudotojo patirtį. Tačiau, izoliuotas tapatybės valdymas pasižymi nuoseklia vartotojo sąsaja (dėl visų tapatybės valdymo procesų įgyvendimo tame pačiame paslaugų tiekėjo puslapyje), tad tai šiek tiek pagerina bendrą naudotojo patirtį.

1.4.2. Centralizuotas tapatybių valdymas

Modelis

Centralizuotame skaitmeninių tapatybių valdyme egzistuoja vienas tapatybės tiekėjas, į kurį kreipiasi visos paslaugos, esančios to paties paslaugų tiekėjo domene [**Josang2005**]. Kai paslaugų tiekėjui reikia autentifikuoti naudotoją (ar atlikti kitą tapatybės valdymo procesą), jis persiųs naudotojo pateiktus atpažinimo duomenis tapatybės tiekėjui, siekdamas pabaigti procesą [**Cao2010**]. Naudotojui šiame modelyje užtenka vienų atpažinimo duomenų, su kuriais jis gali prisijungti prie visų to paties paslaugų tiekėjo paslaugų. Modelio veikimas iliustruotas 3-iame paveiksle.



3 pav. Centralizuotas skaitmeninės tapatybės valdymas [**Cao2010**]

Centralizuotame modelyje paslaugų tiekėjo ir tapatybės tiekėjo funkcijos tampa atskirtos - tapatybės tiekėjas rūpinasi naudotojo identiteto valdymu, o paslaugų tiekėjas koncentruojasi į paslaugos vystymą. Tai sudaro patrauklesnes sąlygas naudotojui, tačiau taip pat sukuria vieno nesekmės taško (angl. *single point of failure*) sistemą. Tapatybės tiekėjo sistemai tapus nepasiekiamai, naudotojai nebegali naudotis nei viena paslauga tame pačiame domene.

Realizacijos bei įgalinančios technologijos

Centralizuotas modelis tinkamiausias naudoti darbuotojams įmonės ribose arba vieno paslaugų tiekėjo paslaugoms [**Josang2005**]. Pateikiami pavyzdžiai abiems šioms realizacijoms.

Viena iš įmonėse naudojamų realizacijų centralizuotam tapatybės valdymui - katalogų prieigos protokolas (angl. *Lightweight Directory Access Protocol*, toliau LDAP), naudojamas pasiekti ir palaikyti informaciją interneto tinkle [**Strictest2011**]. Šis protokolas dažniausiai sujungiamas su aktyviąja direktorija (angl. *active directory*) ir leidžia laikyti kompanijos darbuotojų tapatybės informaciją vienoje vietoje. Taikomosios programos siunčia užklausas į LDAP serverį, kuriose nurodo norimą atlikti veiksmą (pvz. naudotojo autentifikavimą ar naudotojo atributų atnaujinimą). Informacija per LDAP perduodama LDAP duomenų apsikeitimo formatu (LDIF).

LDAP grįstas vienkartinis prisijungimas leidžia įmonės darbuotojams vieną kartą prisijungti prie tam tikros įmonėje naudojamos programos ir nebekartoti prisijungimo kreipiantis į kitą programą. Tačiau, tai galios tik toms programoms, kurios pasiekiamoms darbuotojams per vidinį intranetą [**Strictest2011**]. Dėl šios priežasties LDAP grįstas centralizuotas tapatybės valdymas retai sutinkamas už įmonių intraneto ribų [**Strictest2011**].

Centralizuotas tapatybės valdymas taip pat gali būti realizuotas ir ne įmonės ribose, jei konkretus paslaugų tiekėjas turi keletą paslaugų, skirtų naudotojams. Tokiu atveju jis gali turėti centralizuotą posistemę tapatybės valdymui, o naudotojui užtenka turėti vieną paslaugų tiekėjo paskyrą visoms įmonės paslaugoms. Tokios realizacijos pavyzdys - „Atlassian“ įmonės paslaugos. Naudotojui pakanka turėti vieną „Atlassian“ paskyrą ir jis gali naudotis skirtingais šio paslaugų tiekėjo produktais, tokiais kaip „Jira“, „Confluence“, „Bitbucket“ bei kitais. Vieną kartą prisijungus prie „Atlassian“ programos (pvz. „Jira“), naudotojas tampa autentifikuotas ir kituose „Atlassian“ tinklalapiuose.

Naudotojų poreikių įgyvendinimas

Iš naudotojo perspektyvos, centralizuotas modelis yra patogesnis nei izoliuotas. Naudotojui pakanka turėti vienus atpažinimo duomenis, kurie bus tinkami visoms konkretaus paslaugų tiekėjo programoms. Tačiau, norint pasiekti kito paslaugų tiekėjo paslaugą, naudotojo turima paskyra nebebus tinkama.

Centralizuotas tapatybės valdymas ne intranete gali patirti sukčiavimo (angl. *phishing*) ataką, jei paslaugų tiekėjas nukreipinėja naudotoją į tinklalapį kitame domene. Tačiau, kadangi paslaugų tiekėjas tapatybės valdymą realizuoja pats, jis gali leisti naudotojui vesti identifikavimo duomenis ir pačiame paslaugos puslapyje (o ne nukreipiant į kitą sistemą) arba nukreipti į tame pačiame domene esantį, paties paslaugų tiekėjo valdomą puslapį. Tai sumažina sukčiavimo jautrumo (angl. *phishing susceptibility*) galimybę.

Naudotojai neturi didelės asmens duomenų kontrolės centralizuotame tapatybės valdyme. Nors, skirtingai nei izoliuotame modelyje, jie suteikia duomenis mažesniam kiekiui taikomųjų programų (nebe kiekvienai programai, o kiekvienam paslaugų tiekėjui), tačiau tai vis dar nėra ideali situacija. Naudotojams vistiek reikia kontroliuoti visas skirtingų paslagų tiekėjų paskyras ir žinoti su jomis susijusias duomenų saugojimo bei platinimo taisykles. Taip pat, jeigu modelis taikomas ne įmonės intranete (kur duomenų apsikeitimui apibrėžtas formatas, pvz. LDIF), naudotojas nežino, kokiu būdu jo prisijungimo ar kiti duomenys bus perduodami iš vienos paslaugos į kitą.

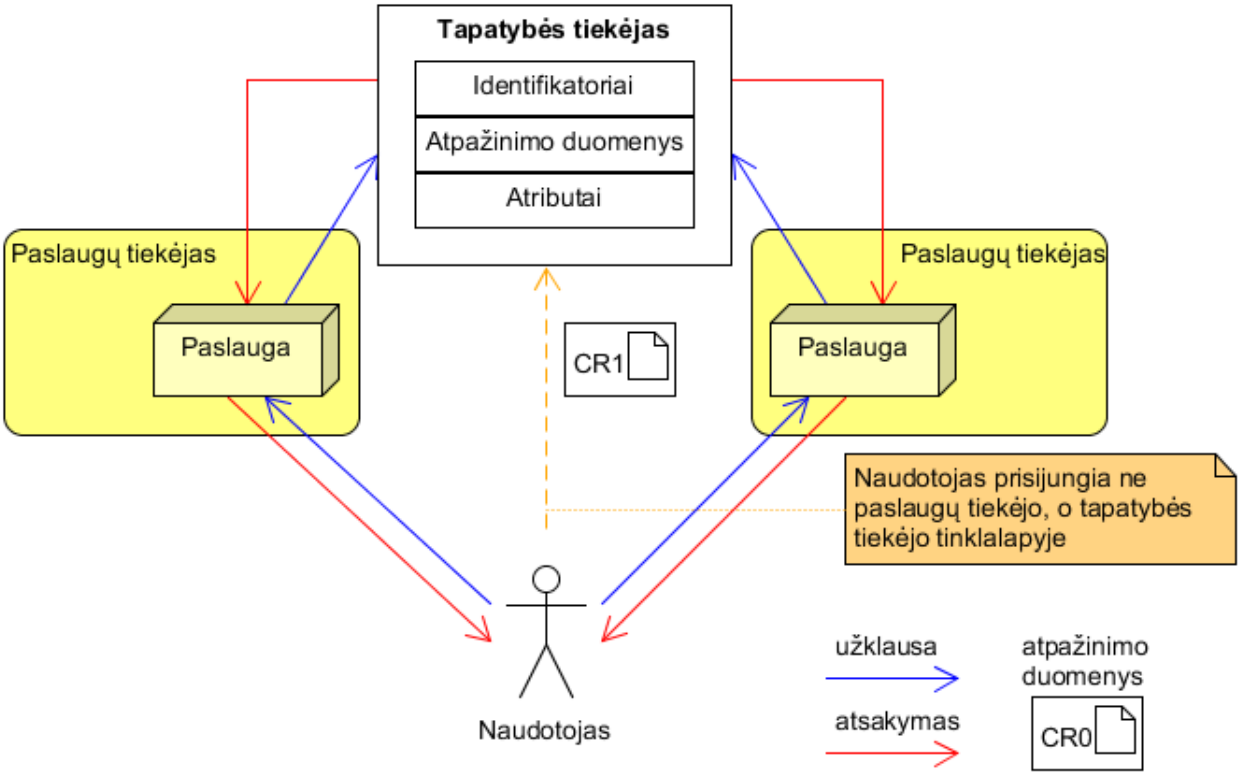
Centralizuotas tapatybės valdymo modelis sudaro palankias sąlygas gerai naudotojo patirčiai užtikrinti. Centralizuotas modelis, priklausomai nuo realizacijos, gali palaikyti vienkartinį prisijungimą, o tai leidžia naudotojui prisijungti vieną kartą ir tapti autentifikuotu visose paslaugų tiekėjo sistemose. Taip pat, kadangi tiek tapatybės valdymo, tiek paslaugų puslapiai yra valdomi paties paslaugų tiekėjo, gali būti užtikrintas vientisas tinklalapių stilius. Taigi, centralizuotą tapatybės valdymą įgyvendinę paslaugų tiekėjai gali sukurti patogias, naudotojams draugiškas (angl. *user-friendly*) sistemas.

1.4.3. Jungtinis tapatybių valdymas

Modelis

Ilgą laiką centralizuoto tapatybių valdymo pakako įmonėms turėti patogų, pačios įmonės prižiūrimą tapatybės valdymo sprendimą. Tačiau augant naudojamų taikomųjų programų bei integracijų su trečiųjų šalių aplikacijomis kiekiui, reikėjo sprendimo, leidžiančio identiteto valdymo uždavinius spręsti ne tik vienos organizacijos ribose. Todėl buvo pradėtas naudoti jungtinis (angl. *federated*) tapatybių valdymas.

Jungtinis (angl. *federated*) tapatybių valdymas yra aibė technologijų ir procesų, kurie leidžia sistemoms dalintis tapatybės informacija ir deleguoti tapatybės valdymo užduotis tarp skirtingų paslaugų tiekėjų [**Maler2008**]. Šis tapatybių valdymo modelis įgalina naudotojus turėti vienus atpažinimo duomenis, kuriuos gali naudoti skirtingų paslaugų tiekėjų tinklalapiuose. Žemiau pateikiama schema, vaizduojanti šio modelio architektūrą:



4 pav. Jungtinis skaitmeninės tapatybės valdymas [**Cao2010**]

Jungtiniame tapatybės valdyme tapatybės tiekėjas yra atskira sistema, su kuria turi integruotis paslaugų tiekėjas. Tapatybės duomenys bei su tapatybe susiję veiksmai (autentifikavimas, autorizavimas) yra deleguojami šiai sistemai. Naudotojas turi vieną identifikatorių, su kuriuo prisijungia tiesiogiai tapatybės tiekėjo puslapyje. Prisijungus šioje sistemoje, naudotojas tampa autentifikuotas visose paslaugose, kurios palaiko šį tapatybės tiekėją [**Maler2008**].

Realizacijos bei įgalinančios technologijos

Kadangi jungtiniame tapatybės valdyme tapatybės tiekėjas bei paslaugų tiekėjas yra skirtingų kūrėjų sistemos, duomenų apsikeitimui tarp jų sukurti technologiniai standartai. Trys šiuo metu labiausiai paplitę protokolai: „SAML“, „OAuth“ bei „OpenID“ [**OIDvsOAuthvsSAML**]. Šių technologijų apžvalga pateikiama 1-oje lentelėje.

1 lentelė. Jungtinio tapatybės valdymo technologijos

	SAML	**OAuth**	**OpenID**
Dabartinė versija	SAML 2.0	OAuth 2.0	OpenID Connect
Paskirtis	Autentifikavimas, autorizavimas, atributų perdavimas	Autorizavimas	Autentifikavimas
Duomenų perdavimas	HTTP, SOAP	HTTP, REST	HTTP, REST
Duomenų formatas	XML	JSON, JWT	JSON, JWT
Duomenų šifravimas	Yra	Yra	Yra
Tapatybės tiekėjo suteiktų duomenų validavimas	Viešo-privataus rakto infrastruktūra	Neapibrėžta (palikta realizacijai)	Viešo-privataus rakto infrastruktūra
Naudotojo sutikimas perduoti duomenis	Nėra	Yra	Yra
Mobiliųjų programėlių palaikymas	Nėra	Yra	Yra
Naudojančios organizacijos	Salesforce, PingFederate, Oracle Access Manager	Google, Amazon, GitHub	Google, Microsoft, Ping Identity

Kiekviena iš minimų technologijų šiandien yra gana plačiai naudojama internete - jų svarbą parodo ir didžiųjų kompanijų („Google“, „GitHub“, „Microsoft“) sprendimai pritaikyti jas savo programinėje įrangoje. Pagrindinis šių standartų skirtumas - jų panaudojimo apimtis. „SAML“ pritaikytas visiems tapatybės valdymo veiksmams, „OAuth“ skirtas naudotojui autorizuoti trečiąją šalį pasiekti jo atributus tapatybės tiekėjo platformoje, o OpenID skirtas naudotojų autentifikavimui.

Naudotojų poreikių įgyvendinimas

Šis tapatybių valdymo modelis yra gana patogus naudotojams. Jis išsaugo centralizuoto modelio privalumus bei išplečia jo funkcionalumą už vienos organizacijos ribų [**OIDvsOAuthvsSAML**]. Naudotojams užtenka turėti vienus atpažinimo duomenis, su kuriais gali prisijungti prie skirtingų paslaugų tiekėjų tinklalapių. Taip pat jungtinis tapatybės valdymas turi vienkartinį prisijungimą, kurį tinklalapyje matyti nori 77% interneto naudotojų [**SSOResearch**]. Vienkartinio prisijungimo pagalba naudotojui užtenka vieną kartą prisijungti savo tapatybės tiekėjo puslapyje ir kreipiantis į visas šį tiekėją palaikančias paslaugas, jam iš naujo prisijungti nebereiks.

Asmens duomenų saugumas priklauso nuo bendravimo tarp paslaugų bei tapatybės tiekėjų, o šiame modelyje jis sudėtingesnis nei izoliuotame ar centralizuotame tapatybių valdyme, nes jungtinis valdymas paremtas tarpdomeniniu bendravimu [**Maler2008**]. Saugumas sustiprinimas

persiunčiamus duomenis pasirašant taikant viešus ir privačius raktus bei užšifruojant. Tačiau, kol dauguma tinklalapių remiasi slapyvardžiu ir slaptažodžiu autentifikuojant naudotoją, šis modelis išlieka labai pažeidžiamas sukčiavimo (angl. *phishing*) atakoms [**Maler2008**].
Atliktas ne vienas tyrimas siekiant nustatyti jungtiniame tapatybės valdyme naudojamų protokolų saugumą [**SAMLSecurity**; **OAuthSecurity**; **OIDCSecurity**]. Pripažinta, kad sukurti automatinį protokolo saugumo analizės įrankį yra gana sunku dėl skirtingų galimų protokolų įgyvendinimo tėkmių [**OIDCSecurity**]. Todėl jungtiniame tapatybės valdyme naudojamų technologijų saugumas stipriai priklauso nuo konkrečių paslaugų bei tapatybės tiekėjų realizacijų detalių. Dažniausios atakos: prieigos rakto vagystės (angl. *token theft*), apsimetimas naudotoju, sesijų apkeitimas ir specifiniai XML formato išnaudojimai, tokie kaip parašų įvyniojimas (angl. *signature wrapping*) [**SAMLSecurity**; **OAuthSecurity**; **OIDCSecurity**]. Šios atakos dažniausiai įgyvendinamos taikant įprastus interneto programišių metodus: XSS (angl. *cross site scripting*, SQL įterpimą, puslapių apgavystes (angl. *phishing*) [**OIDCSecurity**].

Naudotojo duomenų kontrolė jungtiniame modelyje turi tiek privalumų, tiek trūkumų. Viena vertus, naudotojas savo asmens duomenis suteikia mažesniam kiekiui sistemų (tik tapatybės tiekėjams, vietoj visų paslaugų tiekėjų). Tačiau taip paslaugų tiekėjas tampa vieninteliu nesekmės tašku (angl. *single point of failure*) - programišiams įsilaužus į tapatybės tiekėjo sistemą, asmens paskyros visose paslaugose tampa prieinamos [**Pashalidis2003**]. Taip pat, naudotojui gali būti sunku kontroliuoti savo duomenų sklaidą tarp tapatybės tiekėjo ir skirtingų paslaugų, ką rodo ir *Cambridge Analytica* incidentas [**CambridgeAnalytica**]. Nors „OpenID Connect“ bei „OAuth 2.0“ standartai suteikia galimybes naudotojui išreikštinai patvirtinti, kokius duomenis jis sutinka perduoti trečiosios šalies paslaugai, kai jis pradeda ja naudotis (pavyzdys pateikiamas priede nr. 1), tačiau tapatybės tiekėjai ne visada tinkamai informuoja, kokie naudotojo veiksmai gali neišreikštai suteikti leidimą tapatybės tiekėjui perduoti duomenis kitai sistemai.

1.4.4. Skaitmeninės tapatybės valdymo modelių palyginimas

Apžvelgus skirtingas skaitmeninės tapatybės valdymo architektūras, 2 lentelėje pateikiamas modelių palyginimas. Skirtingi modeliai lyginti pagal 1.3 skyrelyje apibrėžtus naudotojų poreikius.

Atsižvelgus į palyginimo rezultatus, galima teigti, kad jungtinis tapatybės valdymas naudotojams yra parankesnis nei centralizuotas ar lokalizuotas valdymas. Šie modeliai verčia naudotojus kurti ir administruoti naujas paskyras nebandytose sistemose, prisiminti daugybę slapyvardžių bei slaptažodžių (ar silpninti saugumą naudojant tą patį slaptažodį) bei kartoti prisijungimo žingsnius, o tai įkyri naudotojams. Pasak kompanijos „Blue Research“ tyrimo, tinklalapiui prašant kurti naują paskyrą 54% interneto naudotojų teigia paliksiantys ar negrįšiantys į tokį puslapį, o 26% ieškos kito panašią paslaugą teikiančio tinklalapio, nereikalaujančio naujos paskyros.

2 lentelė. Skirtingų tapatybės valdymo modelių palyginimas pagal naudotojo poreikius

		Izoliuotas	**Centralizuotas**	**Jungtinis**
Atpažinimo duomenys		Kiekvienai paslaugai	Kiekvienam paslaugų tiekėjui	Kiekvienam tapatybės tiekėjui
Patogumas	Prisijungimų kiekis	Tiek, kiek paslaugų	Tiek, kiek paslaugų tiekėjų	Tiek, kiek tapatybės tiekėjų
	Naudotojo patirties vientisumas	Yra	Yra	Nėra
Saugumas	Nukreipimai (angl. redirects)	Nėra	Galimi	Yra
	Vienas nesėkmės taškas	Nėra	Paslaugų tiekėjo tapatybės valdymo modulis	Tapatybės tiekėjas
	Technologiniai standartai	Paslaugos kūrėjo nuožiūra	Paslaugų tiekėjo nuožiūra	SAML 2.0, Oauth 2.0, OpenID Connect
Kontrolė	Naudotojo patvirtinimas duomenis perduodant kitai paslaugai	-	Nėra	Galimas (Oauth 2.0, OpenID Connect)
	Duomenų suteikimas tik jų prireikus	Tai realizavus paslaugos kūrėjui	Tai realizavus paslaugų tiekėjui	Dalinis (dar nenaudotai paslaugai prašant naudotojo patvirtinimo)
	Tapatybės duomenų keitimas	Kiekvienos paslaugos tinklalapyje	Kiekvieno paslaugų tiekėjo tinklalapyje	Kiekvieno tapatybės tiekėjo tinklalapyje

1.5. Naudotojų problemų tapatybės valdyme apibendrinimas

Skaitmeninės tapatybės valdymas taikant jungtinį modelį išsprendžia nemažai problemų: sumažintas naudotojams reikalingų paskyrų kiekis, įsimintinų slapyvardžių ir slaptažodžių gausa, apribotas reikiamų prisijungimų kiekis. Nepaisant to, nemažai problemų naudotojams išlieka. Pagrindinės iš jų:

- vienintelio nesekmės taško situacija. Jungtiniame tapatybės valdyme tapatybės tiekėjas yra vienintelis nesekmės taškas (angl. *single point of failure*). Tapatybės tiekėjui tapus nepasiekiamam, asmuo internete nebegali naudoti visų paslaugų, kurios buvo integruotos su šiuo tapatybės tiekėju.
- kontrolės trūkumas. Organizacijos renka didžiulius kiekius asmens duomenų - „Facebook“ nuo sukūrimo jau sukaupė 300 petabaitų duomenų apie naudotojus [**Vagata2014**], tačiau naudotojai nėra pakankamai informuoti, kur perduodami jų duomenys. Nors naudojamos technologijos, tokios kaip „OAuth“, išreikštinai prašo naudotojų sutikimo perduodant duomenis naujai paslaugai (žr. priedą nr. 1), tai atliekama tik pradedant naudotis tam tikra paslauga. Taip pat, *CambridgeAnalytica* incidentas [**CambridgeAnalytica**] rodo, kad šio sutikimo gali neužtekti. Visa tai veda prie to, kad naudotojai jaučiasi nekontroliuojantys savo asmens duomenų internete, norėtų lengviau keisti bei trinti juos, o dalį tapatybės valdymo sistemų naudoja tik todėl, kad neturi kitos išeities [**Baars2016**];
- pasitikėjimo stoka. Naudotojams jungtiniame tapatybės valdyme tenka pasikliauti tiek tapatybės tiekėjų, tiek paslaugų tiekėjų gebėjimu patikimai saugoti ir perduoti tapatybės duomenis. Norint naudotojų pasitikėjimo, sistemos turėtų būti skaidrios (angl. *transparent*) apie tai, kaip jos saugo, valdo tapatybės duomenis [**Baars2016**]. Tačiau, kai technologijų specifikaci-

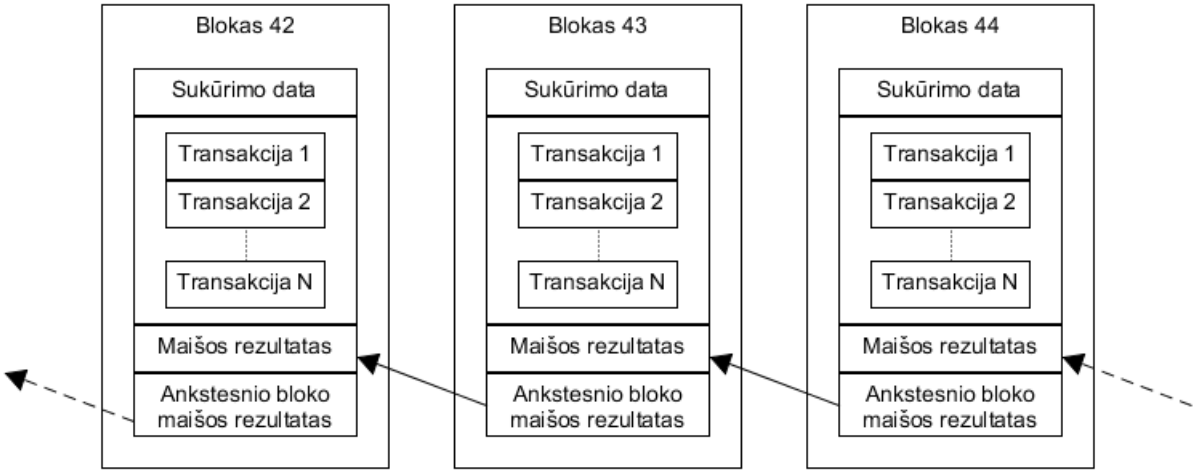
jos palieka vietos nesaugiam protokolų įgyvendinimui [**OAuthSecurity**], o paslaugų tiekėjai retai atskleidžia sistemos veikimo detales [**Baars2016**], naudotojų pasitikėjimas senka;

- saugumo iššūkiai. Dėl naudojamų nukreipimų jungtinis tapatybės valdymas yra itin jautrus sukčiavimo (angl. *phishing*) atakoms, tačiau paslaugų tiekėjai dažniausiai vis dar remiasi naudotojų gebėjimu atpažinti netikrus programišių tinklalapius, nors to nerekomenduoja tyrimai [**Sun2011**]. Taip pat, jungtiniame tapatybės valdyme atsiranda vienas didelės vertės taikinys (angl. *single high value target*) - tapatybės tiekėjas. Įsilaužus į jo sistemą, naudotojų atpažinimo duomenys gali būti pavogti ir panaudoti daugybėje skirtingų paslaugų tiekėjų sistemų.

Pateikti jungtinio tapatybės valdymo trūkumai rodo, kad tapatybės valdymo sritis vis dar yra tobulintina naudotojų atžvilgiu. Naudotojams trūksta didesnės tapatybės duomenų kontrolės, o dėl galimų saugumo iššūkių, vieno nesekmės taško situacijos ir neatskleidžiamų sistemų įgyvendinimo detalių, pasitikėti paslaugų bei tapatybės tiekėjais yra sudėtinga. Toliau darbe apžvelgiama blokų grandinės technologija, atkreipiant dėmesį į jos potencialą spręsti įvardytas problemas skaitmeniniame tapatybių valdyme: vieno nesėkmės taško situaciją bei kontrolės, skaidrumo ir pasitikėjimo trūkumą.

2. Blokų grandinės technologija

Blokų grandinė - tai vieno su kitu susijusių blokų grandinė, kurios blokuose saugomi nekeičiami įrašai [**SatoshiNakamoto**]. Šią technologiją galima apibūdinti kaip daugybę paskirstytų nekintamų skaitmeninių įrašų (angl. *immutable distributed ledger*), tarpusavyje susietų taikant kriptografiją (blokų grandinės pavyzdys pateikiamas 5 paveiksle). Technologija geriausiai žinoma dėl jos panaudojimo „Bitcoin“ kriptovaliutoje. Šiame skyriuje apžvelgiami pagrindiniai blokų grandinės techniniai aspektai, savybės bei galimi skirtingi variantai.



5 pav. Supaprastintas blokų grandinės modelis

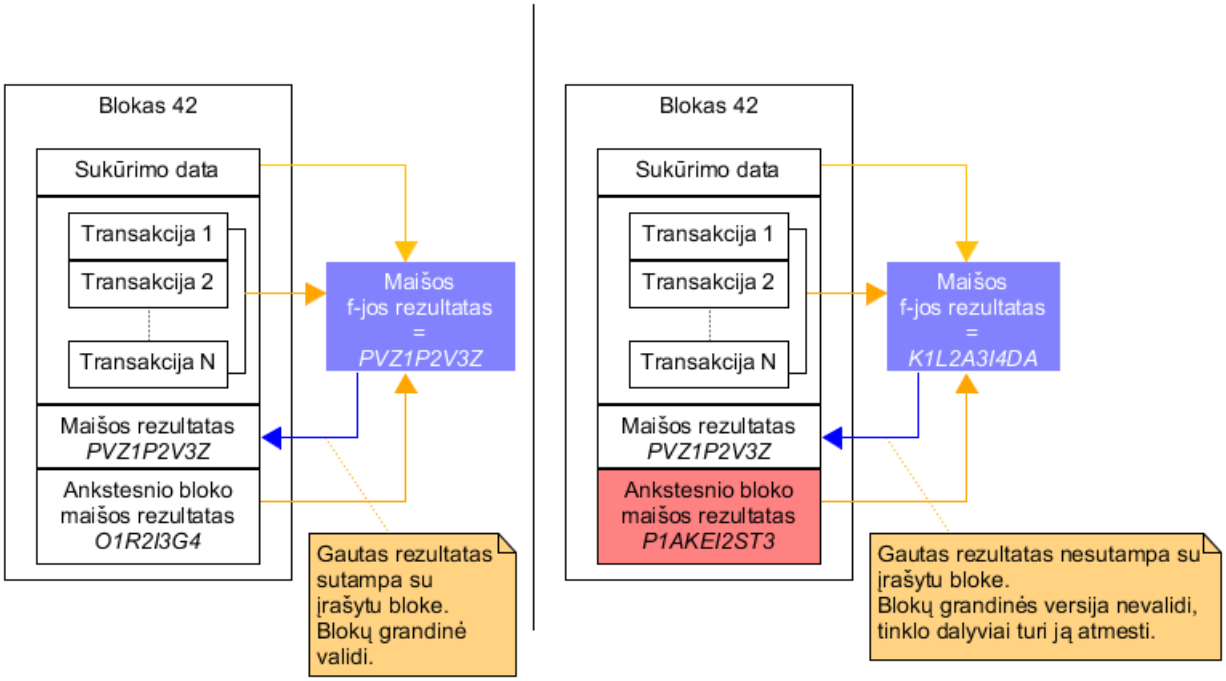
2.1. Nekintamumas

Blokų grandinėje kiekvienas blokas yra sudarytas iš šių dalių:

- transakcijų. Kiekviena transakcija yra duomenys, kuriuos norima saugoti blokų grandinėje. Šie duomenys gali būti bet kokia vertinga informacija: finansinės transakcijos, programinis kodas, asmens duoti sutikimai (angl. *consents*) ar kt. Kiekviena transakcija yra pasirašoma kūrėjo privačiu raktu. Vienas blokas gali turėti vieną arba daugiau transakcijų;
- bloko kriptografinės maišos funkcijos rezultato (angl. *hash*);
- ankstesnio (tėvinio) bloko kriptografinės maišos funkcijos rezultato;
- bloko sukūrimo laiko. Blokai grandinėje saugomi chronologiškai;
- kitų metaduomenų (pvz. bloko eilės numerio, blokų grandinės versijos, *nonce* darbo įrodymui).

Kiekvieno bloko maišos funkcijos rezultatas priklauso nuo jo transakcijų, prieš tai buvusio bloko maišos rezultato ir bloko metaduomenų. Jeigu betkurio bloko duomenys būtų pakeisti, tuomet maišos funkcija sugeneruotų kitokį maišos rezultatą ir būtų lengva patikrinti, kad naujai perskaičiuotas maišos rezultatas nesutampa su bloke esančiu rezultatu. Taip pat, kadangi kiekvienas blokas priklauso nuo prieš tai buvusio bloko, net ir pakeitus vieną iš pirmųjų blokų, pakeitimas

būtų pastebimas pridedant naujus blokus ir būtų galima suprasti, kad turima blokų grandinės versija yra nevalidi (žr. 6 pav.). Tokiu būdu kiekvienas blokų grandinės blokas patvirtina prieš tai buvusio bloko integralumą, taip pasiekiant blokų grandinės nekintamumą (angl. *immutability*), nes perrašyti įrašus blokuose nepastebėtam labai sunku [**SatoshiNakamoto**].



6 pav. Bloko grandinėje validavimas

2.2. Decentralizuotumas

Blokų grandinės sistema yra decentralizuota - nėra vieno centrinio serverio, kuris vienas turėtų visą blokų grandinę. Sistemą sudaro daugybė blokų grandinės mazgų (angl. *node*), kurie turi visą blokų grandinės kopiją. Šie mazgai yra atsakingi už naujų transakcijų validavimą, blokų su transakcijomis kūrimą, sukurtų blokų priėmimą į blokų grandinę ir pranešimus kitiems mazgams apie naują į grandinę priimtą bloką [**Antonopoulos2016**]. Kiekvienas mazgas yra susietas su keletu kitu mazgų. Mazgas, kuris nori pridėti naują bloką (vadinamas *kasėju*), praneša apie jį kitiems mazgams, jie savo ruožtu žinią perduoda kitiems mazgams ir taip ilgainiui kiekvienas mazgas tinkle turi naujausią blokų grandinės versiją.

Kadangi nėra centrinės institucijos, kuri nuspręstų, ar siūlomas blokas yra tinkamas priimti į grandinę, sprendimą bendrai turi priimti visi tinklo dalyviai. Egzistuoja skirtingos taisyklės, vadinamos konsensuso strategijomis (plačiau apie juos 2.4 skyrelyje), kuriomis remdamiesi tinklo mazgai nusprendžia, ar pasiūlytas blokas yra validus. Šios taisyklės apibrėžia, kaip tinklo dalyviai turi įrodyti bloko validumą jį siūlydami į grandinę bei kaip patikrinti kito dalyvio pasiūlyto bloko validumą.

2.3. Tipai

Priklausomai nuo tinklo dalyviams suteikiamų blokų grandinės skaitymo ir rašymo teisių, išskiriami trys pagrindiniai blokų grandinės tipai: vieša, konsorciumo bei privati. Tipų skirtumai pateikiami 3 lentelėje.

3 lentelė. Viešos, konsorciumo ir privačios blokų grandinės palyginimas [**Zheng2017**]

	Vieša	**Konsorciumo**	**Privati**
Konsensuso nustatymas	Visi kasėjai	Išrinkti tinklo dalyviai	Viena organizacija
Skaitymo teisės	Viešos	Gali būti viešos ar apribotos	Gali būti viešos ar apribotos
Centralizuotumas	Nėra	Dalinis	Yra
Efektyvumas	Mažas	Didelis	Didelis

Kadangi vieša blokų grandinė yra atvira visam pasauliui, visiems matomos ir joje išsaugotos transakcijos. Tai sudaro puikias salygas įrašų auditui, tačiau sumažina naudotojų privatumą. Siekiant išlaikyti tam tikrą privatumo lygį, viešoje grandinėje matomi tik transakcijas atlikusių asmenų vieši raktai [**SatoshiNakamoto**].

Privačios bei konsorciumo blokų grandinės yra tik dalinai decentralizuotos - jose blokų validavimą ir priėmimą į grandinę atlieka vienas ar dalis tinklo dalyvių. Šios grandinės privalumai: visi validuotojai yra žinomi, grandinės efektyvesnės dėl greičiau priimamų blokų, apribotos blokų skaitymo teisės suteikia didesnį privatumo lygį, o iškilus poreikiui, tinklo dalyviai gali pakeisti ar atšaukti įvykusias transakcijas [**Buterin2015**]. Konsorciumo ir privačios blokų grandinės labiau tinkamos įmonių vidiniam (ar jungtiniam, pvz. tarp kelių finansinių institucijų) naudojimui. Blokų grandinių karkasų sprendimus įmonėms siūlo „IBM“, „Microsoft“, „Amazon“ [**Zheng2017**].

2.4. Konsensuso strategijos

Kadangi blokų grandinės sistema yra decentralizuota, nėra centrinės institucijos, kuri nuspręstų, ar naujai siūlomas pridėti į grandinę blokas yra validus (be transakcijų su falsifikuotais duomenimis). Todėl blokų grandinės tinkle taikoma konsensuso strategija, pagal kurią nusprendžiama, ar pridėti naują bloką į grandinę. Apžvelgiamos trys dažnai naudojamos strategijos: darbo, įtakos bei autoriteto įrodymo.

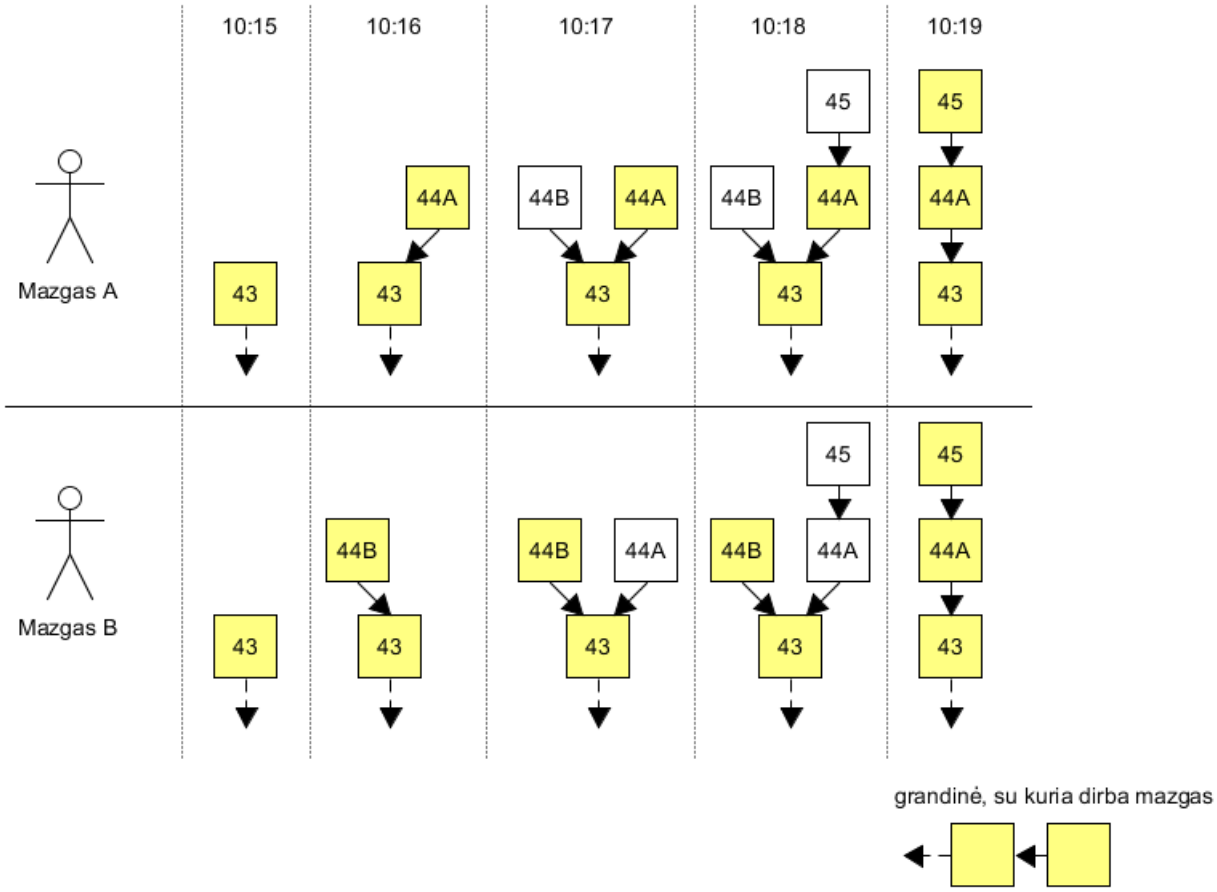
2.4.1. Darbo įrodymo (angl. *proof of work*)

Darbo įrodymo konsensuso strategija remiasi principu, kad daug pastangų ir resursų į bloko validumo įrodymą įdėjęs tinklo dalyvis nebus linkęs sukčiauti. Šioje strategijoje tinklo dalyvis, norėdamas pridėti bloką į blokų grandinę, turi išspręsti laikui ir resursams imlų matematinį uždavinį (užsiima *bloko kasimu*). Pirmas uždavinio reikšmę radęs tinklo dalyvis praneša apie ją kitiems,

kurie turi patvirtinti, ar ši reikšmė teisinga. Jei tai patvirtinta, tinklo dalyviai patikrina, ar naujojo bloko transakcijos yra validžios. Jeigu jos validžios, blokas pridedamas į grandinę [**Zheng2017**].

Darbo įrodymo matematinis uždavinys dažniausiai būna paremtas kriptografine maišos funkcija, kurios rezultatą lengvą validuoti, tačiau duomenis, sugeneravusius šį rezultatą, sunku surasti. Uždavinio tikslas - surasti šiuos duomenis. Tinklo dalyviai eikvoja didžiulius kiekius elektros energijos ir laiko, nes radimas būna paremtas duomenų perrinkimu (angl. *brute force*). Dėl šios priežasties rezultato ieškantiems *kasėjams* neretai būna įvesta paskatinimo sistema, kuri teisingą reikšmę radusį *kasėją* apdovanoja piniginiu atlygiu [**SatoshiNakamoto**].

Kadangi blokų grandinės tinklas yra decentralizuotas, įmanoma situacija, kad labai panašiu metu į grandinę skirtingų mazgų pridėti du validūs blokai. Taip dalis tinklo dalyvių gaus vieną mazgą, o dalis - kitą, o abu jie bus susieti su tuo pačiu prieš tai buvusiu bloku. Tokiu atveju, taikoma ilgiausios grandinės taisyklė (žr. 7 pav.). Mazgai dirba prie pirmiau gauto bloko, tačiau išsaugo kitą gautą bloką kaip šaką. Po to, kai bus gautas dar vienas blokas, jis bus susietas tik su viena iš šakų - taip ši šaka taps ilgesnė. Tuomet ilgesnė šaka paskelbiama aktyviąja grandine, visi su trumpesniąja šaka dirbę mazgai turi pereiti prie aktyviosios grandinės, o atmesto bloko (vadinamo *bloku-našlaičiu*) transakcijos grąžinamos į bendrą transakcijų sankaupą (angl. *transaction pool*) [**SatoshiNakamoto**]. Realiuose blokų grandinės taikymuose, dažnai laukiama keleto iš eilės einančių naujų blokų, kad būtų galima atmesti *bloką-našlaitį*. Pavyzdžiui, „Bitcoin“ blokų grandinėje laukiama apytiksliai 6 blokų iš eilės, kad *bloką-našlaitį* būtų galima atmesti [**Zheng2017**].



7 pav. Ilgiausios grandinės taisyklės taikymas

Pagrindinis šios konsensuso strategijos privalumas yra tas, kad dideli kasimo kaštai gali atgrasyti programišius nuo potencialių atakų. Tačiau, taip veltui išeikvojama daugybė elektros energijos - skaičiuojama, kad kasimas „Bitcoin“ ir „Ethereum“ blokų grandinėms kartu sudėjus per dieną sueikvoja elektros energijos, kurios vertė yra apie 1 milijoną dolerių [**Ethereum**]. Taip pat, dėl ilgo uždavinio sprendimo laiko vieno bloko priėmimas į grandinę gali užtrukti - Bitcoin blokų grandinėje tai užima apie 10 minučių [**Zheng2017**].

2.4.2. Turto įrodymo (angl. *proof of stake*)

Turto įrodymo konsensuso strategija remiasi principu, kad daug blokų grandinės turto turintis kasėjas bus sąžiningas, nes išaiškinus jo nesąžiningumą jis rizikuoja prarasti savo turimą turtą [**Baars2016**]. Šis algoritmas patiki sprendimą priimti tiems tinklo dalyviams, kurie įrodo kad turi daugiausia turto (pvz. blokų grandinės kriptovaliutos). Tai gali pasirodyti kaip nesąžiningas sprendimas, nes turtingiausias tinklo dalyvis gali būti vienvaldžiu sprendimų priėmėju. Dėl to blokų grandinės tinklai neretai taiko šios strategijos variantus: „Peercoin“ papildomai vertina turto amžių, „Blackcoin“ kitą patvirtintoją paskiria pagal atsitiktinę funkciją, kuri atsižvelgia ir į turimą turtą [**Zheng2017**].

Ši strategija leidžia nebeeikvoti didžiulių elektros kiekių, skirtingai nei darbo įrodymo strategija [**Ethereum**]. Algoritmo efektyvumas taip pat sutaupo laiko ir blokai būna greičiau patvirtinami

ir pridedami į grandinę. Tačiau, dėl praktiškai nulinių bloko *kasimo* sąnaudų, galimos dažnesnės tinklo atakos [**Zheng2017**].

2.4.3. Autoriteto įrodymo (angl. *proof of authority*)

Autoriteto įrodymo strategija remiasi keletu tinklo dalyvių, kuriems suteikta teisė validuoti naujus blokus. Ši strategija nebėra tinkama visiškai decentralizuotai blokų grandinei, kurioje būtinas pilnas pasitikėjimo padalinimas [**ProofOfAuthority**]. Tačiau ši strategija tinkama privačioms ar konsorciumo blokų grandinėms.

Šis konsensuso mechanizmas remiasi iš anksto išrinktais tinklo dalyviais, kurie bus atsakingi už blokų validavimą. Kiekvieną kartą pridedant naują bloką į grandinę, vienas iš išrinktų validuotojų patvirtins arba atmes pasiūlytą bloką. Siekiant sumažinti galimą kenksmingų patvirtintojų žalą, įvedamos taisyklės, neleidžiančios tam pačiam validuotojui patvirtinti keleto blokų iš eilės [**ProofOfAuthority**].

Autoriteto įrodymo strategijoje validuotojams svarbu išlaikyti gerą reputaciją - susigadinus ją, validuotojas gali būti pašalintas iš tinklo. Šis konsensuso mechanizmas leidžia greitai ir su itin mažais ištekliais pridėti blokus, tačiau nėra tinkamas pilnai decentralizuotoms blokų grandinėms. Šią strategiją taiko Parity blokų grandinė [**ProofOfAuthority**].

2.5. Išmanieji kontraktai

Išmanusis kontraktas (angl. *smart contract*) - tai kompiuterizuotas transakcijos protokolas, kuris patenkinus numatytus kriterijus išpildo kontrakto sąlygas [**Szabo1997**]. Idėja, pristatyta dar prieš 2000-uosius, šiais laikais yra įgyvendinama pasitelkiant blokų grandinę. Išmanųjį kontraktą galima apibūdinti kaip tam tikras sąlygas (išreikštas kodu), kurios, patenkinus reikalavimus, bus įvykdytos automatiškai, be trečiosios šalies priežiūros.

Blokų grandinės leidžia programų, parašytų su išmaniaisiais kontraktais veikimą, padaryti prieinamu visiems. Patalpinus išmanųjį kontraktą į blokų grandinę, kontrakto kodas taps viešai matomu visiems tinklo dalyviams. Taip kiekvienas norintis gali pamatyti, kaip realizuota programa. Tačiau, kadangi kodas patalpintas į blokų grandinę, jis nebegali būti keičiamas. Tai padidina naudotojų pasitikėjimą, nes jokia centrinė institucija negali pakeisti kontrakto sąlygų, tačiau jei kode palikta defektų, jų nebebus galima ištaisyti.

Blokų grandinėje esančių išmaniųjų kontraktų kodas yra vykdomas automatiškai tinkle esančių kasėjų [**Zheng2017**]. Taip nebelieka vienintelio nesekmės taško situacijos, kai kodą vykdytų vienas centrinis serveris - išmaniojo kontrakto kodas butų nebevykdomas tik tada, jei visi tinklo mazgai išeitų iš tinklo. Už tai, kad tinklo dalyviai naudoja savo kompiuterių resursus išmaniųjų kontraktų kodui vykdyti, jiems dažniausiai suteikiamas piniginis paskatinimas.

Egzistuoja kelios platformos, pritaikytos išmaniųjų kontraktų kūrimui. Bene žymiausia iš jų - „Ethereum“. 2015-aisias pradėjusi veikta platforma pristatė blokų grandinę, kuri pritaikyta kurti būseną galinčius keisti išmaniuosius kontraktus su išbaigta (angl. *turing complete*) programavimo kalba [**EthereumWhitePaper**]. „Ethereum“ kontraktai vykdomi visų tinklo dalyvių, „Ethereum“

virtualiosios mašinos pagalba. Dalyviai už kontraktų kodo vykdymą apdovanojami šios platformos kriptovaliuta - eteriu.
„Ethereum“ apibrėžia dviejų tipų paskyras, skirtas tinklo dalyviams atlikti transakcijas. Išorinės paskyros (angl. *externally owned accounts*), savininkų valdomos privačiais raktais, neturi kodo, o iš jų galima siųsti žinutes kitai išorinei paskyrai sukūrus ir pasirašius transakciją. Kontraktų paskyros (angl. *contract accounts*) turi kodą bei būseną, o priėmusios žinutę iš kitos paskyros gali keisti būseną ar siųsti žinutes kitiems kontraktams.
„Ethereum“ išmanieji kontraktai gali saugoti bei keisti būseną (kontrakto apibrėžtus duomenis) ar kviesti kitų kontraktų funkcijas siunčiant jiems žinutes. Kontrakto vykdymas yra apmokestinamas „Ethereum“ kuru. Kontrakte apibrėžtos funkcijos kvietėjas už kiekvieną vykdytą operaciją, priklausomai nuo jos skaičiavimo sudėtingumo, turi susimokėti tam tikrą kuro kiekį. Kuro sistema padeda apsisaugoti nuo begalinių ciklų vykdymų - jei vykdant funkciją baigiasi kuras, vykdymas nutraukiamas, o kuras nėra grąžinamas [**EthereumWhitePaper**].

2.6. Pavojai ir trūkumai

Blokų grandinės technologija sukuria sąlygas decentralizuotai, nesuteikiant pasitikėjimo vienam ar keliems dalyviams, laikyti nekeičiamus duomenis. Tai atveria įvairių galimybių finansų, daiktų interneto, reputacijos sistemų ar saugumo srityse [**Zheng2017**], tačiau ši technologija turi ir trūkumų, kurių galimą poveikį būtina įvertinti prieš taikant šią technologiją konkrečioje srityje. Pagrindinės grėsmės ir trūkumai aptariami šiame skyriuje.

2.6.1. Daugumos ataka

Viešose blokų grandinėse dauguma (>50%) mazgų tinkle turi patvirtinti bloką, kad jis būtų priimtas į grandinę. Potencialus įsilaužėlis gali pateikti falsifikuotą blokų grandinės bloką (pvz. su netikromis transakcijomis), tačiau kol jis neturi daugumos skaičiavimo galios tinkle, šis blokas bus atmestas likusių dalyvių mazge (ir taps *bloku-našlaičiu* taikant ilgiausios grandinės taisyklę). Tačiau jeigu įsilaužėlis (ar keletas jų) turi daugumą skaičiavimo galios, jis gali dirbti su falsifikuota blokų grandinės versija greičiau negu likę dalyviai tinkle ir taip ilgiausia grandine, prie kurios pereis visi dalyviai, taps jo sukurta grandinė su falsifikuotais blokai [**Zheng2017**]. Ši ataka neištiko dviejų žinomiausių blokų grandinių Bitcoin bei Ethereum, tačiau buvo įvykdyta prieš Verge blokų grandinę [**Sedgwick2018**].

2.6.2. Plečiamumas

Blokų grandinės plečiamumas matuojamas pagal du kriterijus: transakcijų pralaidumą ir saugojimo reikalavimus mazgams. Transakcijų pralaidumas, kuris priklauso nuo to, kaip greitai nauji blokai su transakcijomi yra pridedami į blokų grandinę, susijęs su taikoma konsensuso strategija. Kuo taikoma strategija leidžia greičiau priimti naują bloką, tuo greičiau transakcijos bus patvirtintos. „Bitcoin“ kriptovaliutos blokų grandinėje, taikančioje darbo įrodymo konsensusą,

apdorojoma apie 7 transakcijas per sekundę [**Zheng2017**], kai „Tendermint“ blokų grandinė, taikanti atsparumo klaidoms konsensusą, teigia galinti apdoroti tūkstančius transakcijų per sekundę [**Tendermint2017**]. Dėl darbo įrodymo strategijos neefektyvumo blokų grandinės dažnai keičia ją į kitą konsensuso strategiją - „Ethereum“ blokų grandinė ketina pereiti prie turto įrodymu grįsto konsensuso [**Ethereum**].

Kita priežastis, dėl ko transakcijos tvirtinamos gana lėtai - blokai turi dydžio apribojimus. Dėl šių apribojimų, tik dalis susikaupusių transakcijų gali būti priimtos į naują bloką, o likusios turi laukti, kol pateks į kitą bloką. Tam spręsti pasitelktos šalutinės blokų grandinės. Jos dalį bloko duomenų iškelia į šalutinę grandinę, taip palikdamos daugiau vietos pagrindinės grandinės bloke. Tokį sprendimą priėmė „Bitcoin“ blokų grandinė, į tinklą pristatydama „SegWit“ protokolą, kuris iškelia skaitmeninių parašų duomenis į atskirą grandinę [**Segwit**].

Visos blokų grandinės dydis taip pat gali sukelti plečiamumo problemų. Šiuo metu Bitcoin blokų grandinė užima per 100 gigabaitų [**Zheng2017**]. Siekiant sumažinti blokų grandinės (kurią turi kiekvienas mazgas tinkle) dydį, siūlomi įvairūs sprendimai. Vienas variantas yra mazgams neturėti seniausių blokų grandinės dalių, o senus blokus su transakcijomis iškelti į atskirą duomenų bazę. Kitas sprendimo būdas - pagrindinėje blokų grandinėje laikyti tik transakcijų maišos rezultatą, o pačius transakcijų duomenis iškelti už grandinės ribų (angl. *off-chain*) [**Lo2017**].

2.6.3. Anonimiškumas

Manoma, kad blokų grandinės išsaugo naudotojų privatumą dėl to, kad transakcijose atskleidžiamas tik naudotojų viešas raktos vietoj tikrosios tapatybės [**Zheng2017**]. Tačiau, tyrimai rodo, kad iš viešojo rakto galima atsekti tikrąją naudotojo tapatybę [**Barcelo2007**]. Taip pat, transakcijoje buvę duomenys (pvz. pervestas pinigų kiekis) viešoje blokų grandinėje bus matomas visiems tinklo dalyviams.

Jei reikia didesnio anonimiškumo, tam yra keletas galimų sprendimų. Pirmiausia, reikia atsižvelgti, ar negalima naudoti privačios blokų grandinės, kurioje blokų skaitymo teisės būtų apribotos. Kitas variantas - blokų grandinėje saugoti užšifruotus duomenis, kurių skaitymui reikia turėti dešifravimo raktą. Jeigu tai netinkama išeitis, tuomet transakcijas galima atlikti per tarpininką, kuris išskaidys vieną transakciją į keletą, vykdomų su skirtingais viešaisiais raktais, ir taip bus sunkiau atsekti transakcijos autorių [**Zheng2017**].

2.7. Pritaikymas tapatybės valdyme

Blokų grandinės savybės atveria įdomių galimybių skaitmeninių tapatybių valdyme:

1. Decentralizuota sistema eliminuoja vienintelio nesekmės taško situaciją. Pritaikius decentralizuotą blokų grandinę tapatybės valdymo sistemoje, dėl blokų grandinės išskirstymo po skirtingus mazgus tapatybės tiekėjas nebebūtų vienintelis nesekmės taškas identiteto valdymo infrastruktūroje. Tokiu būdu tapatybės informacija būtų prieinama ir tada, kai tapatybės tiekėjas yra nepasiekiamas.

2. Visiems prieinami grandinės įrašai ir sistemos veikimas padidina skaidrumą. Blokų grandinėje saugant naudotojo suteiktas prieigas skirtingoms paslaugoms, naudotojas betkada galėtų matyti paslaugoms suteiktas teises. Prieigų suteikimą aprašius išmaniuosiuose kontraktuose, paslaugų užklausos duomenimis pasiekti galėtų būti suteiktos tik tada, kai naudotojas jas patvirtina.
3. Nekintamumas apsaugo nuo galimų įsilaužimų siekiant pakeisti įrašus. Blokų grandinėje saugant naudotojo atributus ar paslaugų prieigos teises, o rašymo teises į grandinę turint tik naudotojui ar jo pasirinktoms programoms, naudotojas būtų užtikrintas, kad neįvyko jo paties neautorizuotų pakeitimų.

2.6 skyrelyje įvardyti pagrindiniai blokų grandinės apribojimai nesudaro didelių kliūčių šią technologiją taikyti skaitmeninės tapatybės valdyme. Anonimiškumo klausimą galima spręsti šifruojant saugomus tapatybės duomenis, o raktus dešifravimui turint tik autorizuotoms paslaugoms. Blokų grandinės greitaveika neturėti kelti problemų standartinėms tapatybės valdymo sistemos operacijoms [**Lo2017**]. Daugumos ataka decentralizuotoje viešoje blokų grandinėje išlieka kaip galimas pavojus, tačiau tapatybės sistemai pasirinkus tokią grandinę, kurioje šios atakos vykdymas reikalauja milžiniškų resursų, jos įvykdymo tikimybė išlieka gana maža.

Apžvelgus blokų grandinės technologiją galima teigti, kad jos savybės tinkamos naudotojų problemoms skaitmeniniame tapatybės valdyme spręsti. Blokų grandine paremta decentralizuota, skaidri ir apsauganti nuo neleistino įrašų keitimo sistema gali padėti įveikti kontrolės, pasitikėjimo trūkumo bei vienintelio nesekmės taško iššūkius tapatybės valdyme.

3. Blokų grandine paremtas tapatybės atributų valdymo modelis

Atsižvelgus į blokų grandinės savybes, tinkamas tapatybės valdymo problemoms spręsti, šiame skyriuje pateikiamas blokų grandine paremtas skaitmeninės tapatybės valdymo modelis. Modelis skirtas spręsti 1.5 skyrelyje apibendrintoms naudotojų problemoms. Kadangi tapatybės valdymas apima daug skirtingų procesų, aprašomas modelis apsiriboja asmens tapatybės atributų valdymu ir perdavimu. Kuriamo modelio architektūra remiasi MIT tyrime [**MITPaper**] pristatytais teoriniais protokolais, bandant juos modifikuoti ir pritaikyti ne tik mobiliosioms aplikacijoms, o ir interneto tinklalapiams.

3.1. Reikalavimai

Naudotojams kylančios problemos tapatybės valdyme remiasi į tai, kad jų tapatybė yra visiškai patikėta valdyti tapatybės tiekėjams. Dėl menko naudotojų įsitraukimo į tapatybės valdymo procesus, asmenys neretai lieka tik pasyvūs stebėtojai.

Siekiant išspręsti šią problemą, kuriamo modelio reikalavimai kurti pagal į naudotoją orientuotos tapatybės (angl. *user-centric identity*) principus. Ši paradigma akcentuoja naudotojus kaip centrinę identiteto valdymo sistemų dalį, perduodant paslaugų ir tapatybės tiekėjų turimą tapatybės kontrolę naudotojams [**Cao2010**]. Tokiu būdu, naudotojai turi aktyviau prižiūrėti ir dalyvauti tapatybės valdymo procesuose (dažniausiai naudojant papildomą programinę įrangą), tačiau geriau žino, kaip ir kur yra naudojami jų asmens duomenys.

Taikant į asmenis orientuotos skaitmeninės tapatybės principus, reikalavimai modeliui suformuluoti naudotojų istorijų forma:

1. Kaip interneto naudotojas, aš noriu žinoti, kurios programos turi prieigą prie kurių mano asmens duomenų.
2. Kaip interneto naudotojas, aš noriu kontroliuoti savo asmens duomenis ir pats suteikti arba atmesti prieigą paslaugų tiekėjams pasiekti mano duomenis.
3. Kaip interneto naudotojas, aš noriu galimybės lengvai atnaujinti savo asmens duomenis vienoje vietoje.
4. Kaip interneto naudotojas, aš nenoriu, jog mano asmens duomenų pasiekiamumas priklausytų tik nuo vienos trečiosios šalies pasiekiamumo.

Pateiktos naudotojų istorijos padengia 1 skyriuje apibrėžtus asmenų poreikius tapatybės valdymo sistemoms bei išskirtas kontrolės, pasitikėjimo ir skaidrumo trūkumo problemas. Interneto tapatybių valdymo sistemo saugumo atakos (angl. *cross site scripting*, *phishing*) yra plati tema, verta atskiro tyrimo, tad ji šiame modelyje nebus nagrinėjama. ar užtenka tokio sakinio pasakymui kad čia *out of scope*? nes saugumui užtikrint reik kad naudojamos paslaugos SSL turėtų, ir šiaip plati tema labai, kurios vien blockchain neišspręs

3.2. Modelio dalys

Modelis sudarytas iš trijų dalių: išmaniųjų kontraktų paslaugų autorizavimo logikai bei atributų saugojimui, atributų valdymo programos bei paslaugų registro.

3.2.1. Išmanieji kontraktai paslaugų autorizavimo logikai

Pagrindinė blokų grandinės paskirtis yra pagal išmaniuosiuose kontraktuose aprašytą logiką suteikti arba nesuteikti prašančioms paslaugoms jų norimus tapatybės atributus. Išmanieji kontraktai yra atsakingi už:

- naudotojo atliekamą paslaugų autorizavimą. Naudotojas, kviesdamas išmanaus kontrakto funkciją, gali pasirinkti, ar suteikti konkrečiai paslaugai prieigą prie jos norimo atributo. Prieigos yra valdomos ne visos tapatybės, o atskirų atributų lygmenyje.
- šifruotų atributų saugojimą. Išmaniuosiuose kontraktuose saugomi naudotojo tapatybės atributai. Jeigu paslaugą P naudotojas N yra autorizavęs atributui A, tuomet kontrakte išsaugomas N ir P simetriškų šifro raktu (angl. *symmetric encryption key*) užšifruotas atributas A. Taip tik paslauga P ir naudotojas N galės perskaityti viešame išmaniajame kontrakte esantį atributą.
- pateikiamas funkcijas paslaugoms pasiekti atributus. Išmanusis kontraktas atsižvelgia į tai, kokia paslauga kreipiasi (iš jos adreso) bei kokio atributo prašo. Tuomet, patikrinęs, ar ši paslauga turi prieigą, atitinkamai persiunčia paslaugai atributą, praneša apie atmestą prieigą arba išsaugo užklausą, laukiančią naudotojo sprendimo.

3.2.2. Atributų valdymo programa

Atributų valdymo programa yra skirta palengvinti naudotojo bendravimą su blokų grandine. Teoriškai, naudotojas galėtų pats formuoti užklausas bei sugeneruoti simetriško šifro raktą ir perduoti jį paslaugai, tačiau tai nėra patogu. Dėl šių priežasčių, ši atributų valdymo programa padeda naudotojui matyti bei keisti suteiktas išmaniajame kontrakte esančias prieigas, užšifruoti ir persiųsti atributus į blokų grandinę bei sugeneruoti šifro raktą, unikalų kiekvienai naudotojo N ir paslaugos P porai.

Modelyje nėra apibrėžiama, kaip turi būti įgyvendinta ši programa - tai gali būti kompiuterio darbalaukio programa, mobilioji programėlė ar interneto tinklapis. Jos įgyvendinimas būtų panašus į centralizuotos kriptovaliutos piniginės kūrimą (pvz. „Coinbase“), nes ši programa skirta palengvinti naudotojo bendravimą su blokų grandine, kuris aprašytas 3.2.1 skyrelyje ir stebėti jau suteiktas prieigas.

Atributų valdymo programa saugo naudotojo privatų raktą, praneša apie naujas paslaugų užklausas, padeda šifruoti į blokų grandinę saugomus atributus su paslaugos viešuoju raktu. Taip pat, joje naudotojas matytų suteiktų prieigų apžvalgą ir betkada galėtų pakeisti konkrečių paslaugų pasiekiamus atributus.

3.2.3. Paslaugų registras

Šiame modelyje paslaugos yra identifikuojamos pagal jų blokų grandinės paskyros adresą. Atributų valdymo programa, skaitanti išmaniajame kontrakte laukiančias užklausas, jose mato išsaugotą besikreipusios paslaugos adresą.

Nors adresas yra viešas ir unikaliai identifikuoja kiekvieną blokų grandinės dalyvį, turintį paskyrą, tačiau iš adreso nustatyti tikrąją dalyvio tapatybę yra sudėtinga - pats adresas nėra tiesiogiai susiejamas su tikru asmens identifikatoriumi (pvz. asmens kodu ar paslaugos įmonės kodu). Tokiu būdu, atributų valdymo programoje matant tik paslaugos adresą dalyvis gali nežinoti, kokią paslaugą jis autorizuoja.

Šiai problemai spręsti modelyje yra atskiras išmanusis kontraktas, kuris funkcionuoja kaip *paslaugų registras*. Kiekviena paslauga, besinaudojanti modeliu, atlieka vieną transakciją į šį registrą, nusiunčiant savo vardą ir taip kontraktas išsaugo, iš kokio adreso į jį kreiptąsi (nes transakcija yra pasirašyta kontrakto kvietėjo ir tik jis gali atlikti transakcijas iš jo paskyros). Tuomet, paslaugai viešai atskleidus savo adresą (pvz. pateikus savo tinklalapyje prieigos tašką (angl. *endpoint*), kuris jį grąžina), būtų galima nesunkiai patikrinti, ar paslauga, besiskelbianti adresu X, iš tikrųjų yra užsiregistravusi *paslaugų registre*. Tokiu būdu atributų valdymo programa galėtų kreiptis į *paslaugų registro* išmanųjį kontraktą, patikrinti ar paslauga jame užsiregistravusi ir iš jo sužinoti paslaugos pavadinimą.

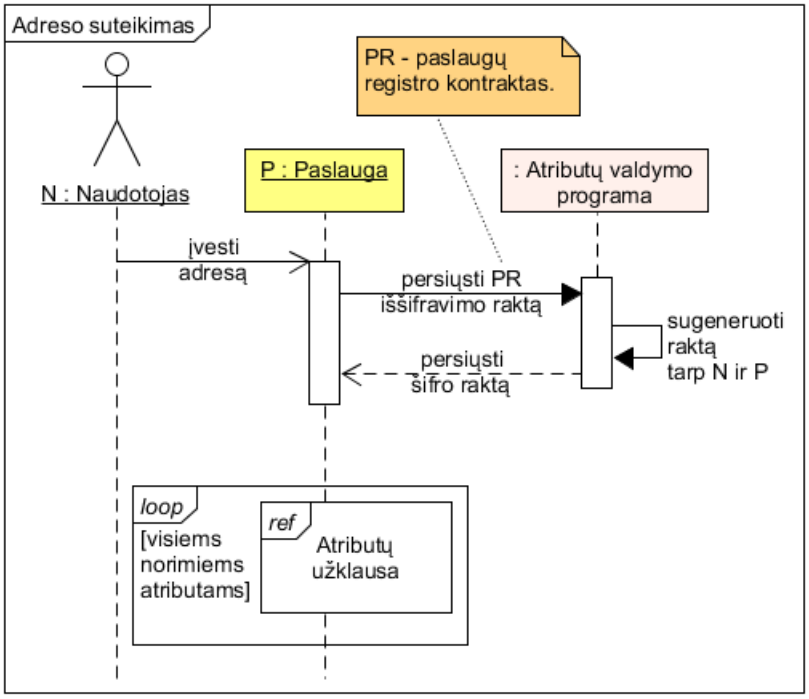
3.3. Pagrindiniai naudojimo atvejai

Šiame skyriuje UMl sekų diagramomis pateikiami pagrindiniai modelio panaudos atvejai (angl. *use cases*).

3.3.1. Naudotojo adreso suteikimas paslaugai

Kai paslauga bei asmuo naudojasi aprašomu modeliu, paslauga paprašo naudotojo pateikti jo adresą (naudotojo patogumui, adresas rodomas atributų valdymo programoje), kuris yra viešas naudotojo identifikatorius blokų grandinėje. Gavus šį adresą, paslauga iš atributų valdymo programos prašo šifro rakto, su kuriuo galės dešifruoti iš blokų grandinės gautus atributus. Su gautu adresu paslauga P gali kreiptis į blokų grandinę ir prašyti norimų naudotojo atributų (žr. 8 pav.).

Modelis neapibrėžia, kada naudotojas N privalo pateikti adresą paslaugai P. Svarbu, kad paslauga P gautų šį adresą ir unikalų šifro raktą tarp N ir P, nes be jų P negalės gauti ir perskaityti norimų tapatybės duomenų. Šis suteikimas galėtų būti atliekamas prisijungimo metu, įtraukiant papildomą žingsnį.



8 pav. Naudotojas adreso suteikimas paslaugai

3.3.2. Paslaugos atliekama atributo užklausa

Paslauga P, norinti gauti tam tikrą asmens atributą (pvz. gimimo datą)[1] , kreipiasi į blokų grandinės išmanųjį kontraktą (žr. 9 pav.). Kontraktas tuomet patikrina, ar ši paslauga turi prieigą prie pageidaujamo atributo. Jei turi, tuomet grąžina šį atributą. Jis užšifruotas paslaugos P turimu šifro raktu, tad paslauga gali jį dešifruoti ir perskaityti duomenis. Jei naudotojas atmetęs prieigą, apie tai pranešama paslaugai. Jei naudotojas dar nesvarstęs šios prieigos, išsaugoma laukianti (angl. *pending*) paslaugos P atributo A užklausa ir laukiama naudotojo sprendimo. Naudotojas šią laukiančią užklausą galės pamatyti savo atributų valdymo programoje ir ten atlikti sprendimą.

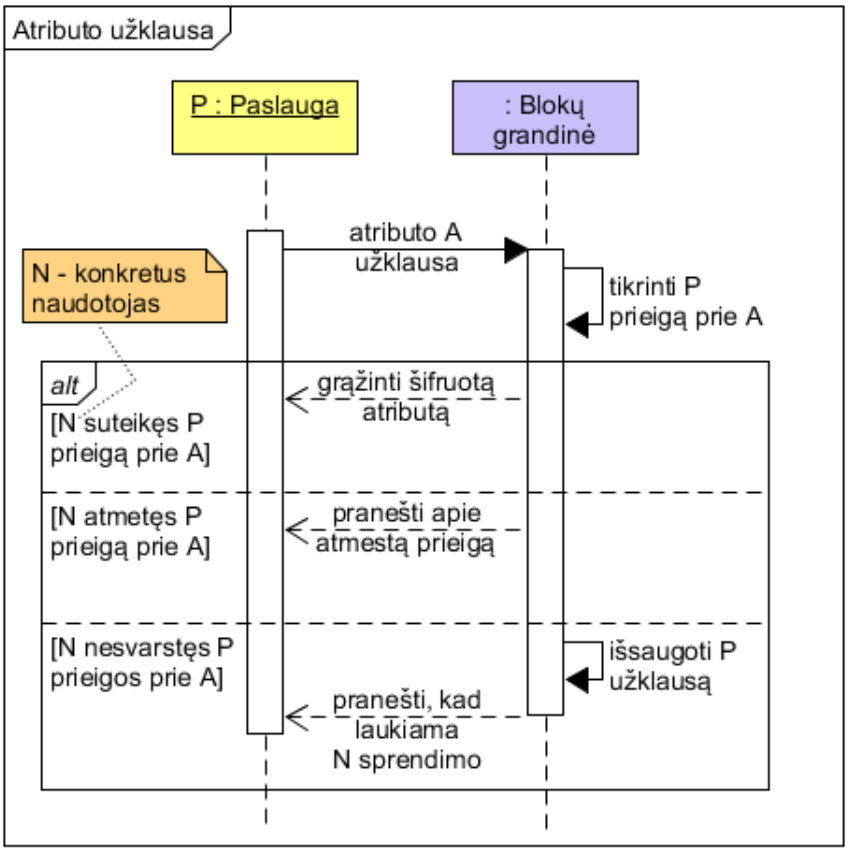
3.3.3. Naudotojo atliekamas paslaugos autorizavimas

Atributų valdymo programa, stebinti išmaniuosiuose kontraktuose naujas paslaugų užklausas, praneša naudotojui apie dar neatliktus autorizavimus. Tuomet naudotojas gali pasirinkti, ar suteikti, ar atmesti prieigą prie norimo atributo A paslaugai P. (žr. 10 pav.). Blokų grandinė asmeniui leidžia autorizuoti tik savo naudojamas paslaugas [2].

Jei naudotojas prieigą suteikia, tuomet atributų valdymo programa užšifruoja atributą A. Tai daroma todėl, kad kiekvienas atributas, suteiktas paslaugai P, turi būti užšifruotas tik paslaugai P ir naudotojui N žinomu šifro raktu bei nusiųstas į išmanųjį kontraktą. Pati atributo reikšmė yra iš

[1] Paslaugai kreipiantis į kontraktą, ji turi žinoti, koks yra norimo atributo identifikatorius, kurį reikia perduoti į kontrakto funkciją. Atributų identifikatoriai gali būti pateikiami atskirame išmaniajame kontrakte arba kuriamo modelio interneto puslapyje.

[2] Ar naudotojo N prieigai pakeisti siunčiamą žinutę tikrai siunčia N, blokų grandinė atskiria iš žinutės parašo.



9 pav. Paslaugos atliekama atributo užklausa

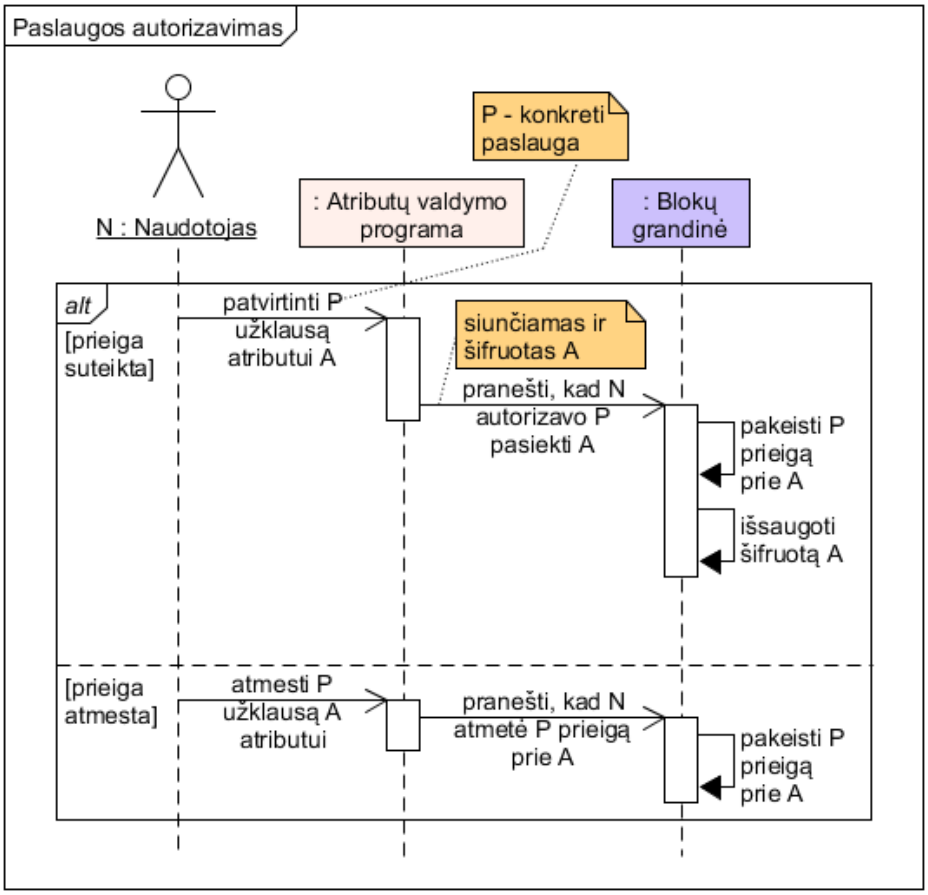
anksto įvesta naudotojo į atributų valdymo programą. Užšifravus reikšmę, duomenys nusiunčiami į išmanųjį kontraktą ir jame pažymima, kad paslauga P autorizuota pasiekti atributą A.

Jei naudotojas prieigą atmeta, tuomet blokų grandinėje pažymima, kad paslauga P nėra autorizuota pasiekti atributą A.

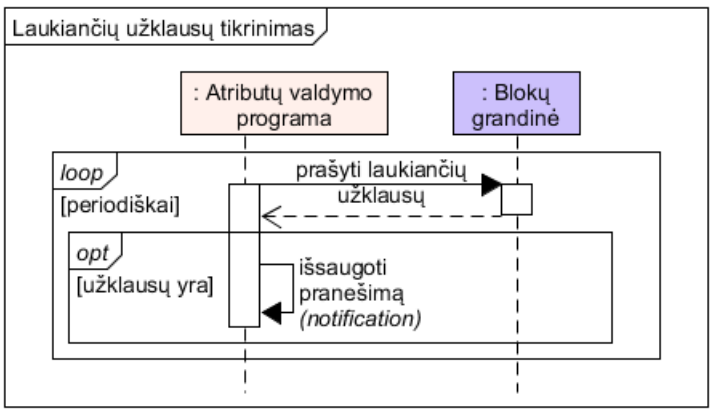
Jei naudotojas vėliau nuspręstų pakeisti savo sprendimą (pvz. panaikinti suteiktas prieigas prie atributo A paslaugai P), jis šį paslaugos autorizavimo procesą galėtų pakartoti ir jau esamai prieigai.

3.3.4. Atributų valdymo programos atliekamas blokų grandinės stebėjimas

Blokų grandinės išmanusis kontraktas negali tiesiogiai pranešti išorinei (ne išmaniajame kontrakte esančiai) programai apie įvykusius pasikeitimus. Todėl išmanusis kontraktas gali saugoti būseną, o ją vėliau pasiekia norinti taikomoji programa. Tokiu būdu atributų valdymo programa periodiškai kreipiasi į išmanųjį kontraktą ir žiūri, ar yra naujų paslaugų užklausų. Jeigu jų yra, atributų valdymo programa išsaugo pranešimą apie tai naudotojui (žr. 11 pav.).
Panašiu stebėjimo principu pasikeitimus išmaniojo kontrakto būsenoje gali stebėti ir paslaugų tiekėjai (pvz. apie pasikeitusias prieigas).



10 pav. Naudotojo atliekamas paslaugos autorizavimas



11 pav. Atributų valdymo programos atliekamas blokų grandinės stebėjimas

4. Tapatybės atributų valdymo modelio vertinimas

Šiame skyriuje vertinamas 3 skyriuje pristatytas blokų grandine paremtas tapatybės atributų valdymo modelis: kokie jo privalumai, trūkumai, pritaikymo barjerai.

4.1. Privalumai

4.1.1. Naudotojui suteikta atributų valdymo kontrolė

Naudotojas su atributų valdymo programa gali autorizuoti paslaugas pasiekti jo atributus, esančius blokų grandinėje. Programoje taip pat galima pamatyti visus įvestus atributus ir paslaugas, kurios yra autorizuotos juos pasiekti. Naudotojas taip pat gali pakeisti anksčiau priimtą sprendimą ir programa atnaujins prieigą blokų grandinėje. Naudotojui nusprendus pakeisti atributo reikšmę, jis tai padaro atributų valdymo programoje, o ši kreipiasi į blokų grandinę ir joje atnaujina esančias reikšmes. Kaip galėtų atrodyti tokia atributų valdymo programa pateikiama priede **??**.
Išvardytos funkcijos padengia 3.1 skyrelio 1, 2 ir 3 reikalavimus.

4.1.2. Tapatybės atributai decentralizuoti

Tapatybės atributus saugant blokų grandinėje, atributai tampa pasiekiami nepriklausomai nuo vienos trečiosios šalies pasiekiamumo. Taip tiek paslaugų tiekėjai, tiek naudotojas gali betkada pamatyti suteiktus atributus tol, kol bent vienas blokų grandinės mazgas yra pasiekiamas. Šis privalumas įgyvendina 3.1 skyrelio 4-ą reikalavimą.

4.1.3. Sumažintas reikiamas pasitikėjimas trečiosioms šalims

Kadangi tapatybės atributai yra saugomi viešoje, decentralizuotoje blokų grandinėje, kurios išmaniųjų kontraktų logika prieinama visiems, naudotojams nebereikia pasitikėti vien tik paslaugų bei tapatybės tiekėjais - jų atributai šiuo atveju priklauso nuo korektiško išmaniųjų kontraktų veikimo.

Verta pastebėti, kad pristatytame modelyje naudotojui vis dar reikia pasitikėti atributų valdymo programa. Šiam pasitikėjimui padidinti ši programa turėtų būti atviro kodo, kuris viešai prienamas. Taip pat, paslaugų tiekėjai parsisiuntę atributą iš blokų grandinės, vis dar gali jį išsisaugoti. Tačiau, galimybė paslaugai pačiai nesaugoti naudotojo duomenų turėtų būti pagrindinė paskata naudotis šiuo modeliu - taip naudotojai labiau pasitikės šia paslauga. Galimas išsisaugojimo pavojus pastebimas ir [**MITPaper**] tyrime, kurio autoriai aptaria ir galimybę neleisti paslaugai pasiekti pačio duomens, o atlikti reikalingus skaičiavimus pačiame tinkle.

4.2. Trūkumai

4.2.1. Atributų reikšmių įvedimas

Norint naudotis aprašytu modeliu, asmuo turi į atributų valdymo programą įvesti norimų atributų reikšmes. Taip daroma todėl, kad naudotojui nereiktų vesti atributo kiekvieną kartą, kai

nori autorizuoti tam tikrą paslaugą. Tačiau, tokiu būdu, naudotojas suteikia savo duomenis dar vienai paslaugai - pačiai atributų valdymo programai. Tai suteikia patogumo (nereik daugybę kartų vesti atributo), tačiau naudotojui reikia pasitikėti atributų valdymo programa. Norint to išvengti, galima būtų naudotojo prašyti įvesti atributą kiekvieną kartą, kai paslauga prašo autorizuoti prieigą prie pasirinkto atributo.

4.2.2. Nepatikimos atributų reikšmės

Paslaugų tiekėjai iš blokų grandinės gauna reikšmes, kurias įvedė pats naudotojas. Tai nesudaro jokių keblumų, kai jų teisingumą gali patikrinti pati paslauga (pvz. atsiųsti žinutę į telefono numerį), tačiau kai to atlikti negalima, paslaugai tenka pasitikėti naudotojo įvesta reikšme.

Tai būtų galima tobulinti, jei šias reikšmes galėtų verifikuoti atsakingos institucijos (pvz. bankas). Tuomet, šios institucijos turėtų teikti papildomą paslaugą, kurioje naudotojui paprašius, jos galėtų atlikti užklausą į blokų grandinę ir patvirtinti, kad atributo reikšmė yra teisinga, o tai būtų išsaugota išmaniajame kontrakte. Tokiu atveju, į blokų grandinę besikreipianti paslauga galėtų matyti, ar šis atributas yra verifikuotas. Panašus tvirtinimo mechanizmas nagrinėjamas [**Baars2016**] tyrime.

4.2.3. Kaina

Blokų grandinės išmaniųjų kontraktų funkcijų kvietimai kainuoja pinigus. Įprastu atveju, funkcijos vykdymą apmoka funkcijos kvietėjas. Aprašytame modelyje kvietimus į blokų grandinę vykdo tiek paslaugų tiekėjas, tiek atributų valdymo programa. Kai atributų valdymo programa gali būti mokama naudotojui, nėra aišku, ar paslaugų tiekėjai būtų pasiryžę mokėti už funkcijų kvietimus.

Vienas iš galimų sprendimų - kviečiant funkciją paslaugai, ją apmokėtų ne paslaugos tiekėjas, o pats kontraktas. Taip kontraktas būtų apmokamas iš naudotojo lėšų, o jos naudojamos paslaugai kviečiant kontrakto funkcijas. Tačiau, tokį mechanizmą turi palaikyti pasirinkta blokų grandinė, taikant jį reiktų apsisaugoti ir nuo galimų pavojingų pakartotinių kvietimų, kurie būtų skirti tik išeikvoti kontrakto lėšas.

kai prototipą pakodinsiu, pažiūrėsiu kiek kainuoja kuro funkcijos, jas gal reiks įmest

4.3. Pritaikymo barjerai

Pagrindiniai sunkumai, kurie kiltų tokį modelį taikant praktikoje, yra galimas paslaugų nenoras bendradarbiauti bei menka blokų grandinės technologijos branda.

Paslaugų tiekėjams, norint pradėti naudoti šį modelį savo paslaugose, reikia susikurti blokų grandinės paskyrą bei suintegruoti savo sistemą su blokų grandine. Tai gali būti palengvinta sukuriant viešai prieinamą programinę įrangą, paruoštą naudoti paslaugų tiekėjams, tačiau paslaugų kūrėjai vis tiek turi nuspręsti naudoti šį modelį. Pagrindinė paskata jiems yra didesnis naudotojų pasitikėjimas suteikiant jiems galimybę kontroliuoti savo duomenis bei pagerintas asmens duomenų pasiekiamumas juos iškėlus į decentralizuotą blokų grandinę.

Blokų grandinė yra vis dar gana nauja technologija. Viešos blokų grandinės yra pažeidžiamos daugumos atakų, nemažai skeptikų teigia, kad pagrindinis technologijos panaudojimas išliks tik kriptovaliutoms, jų plečiamumas (angl. *scalability*) aktyviai tiriamas. Taip pat, 2016-aisiais metais išmaniojo kontrakto „The DAO“ trūkumą išnaudojusi ataka privertė „Ethereum“ blokų grandinę atlikti išsišakojimą (angl. *hard fork*), kuris pakeitė transakcijų istoriją ir grąžino įsilaužėlio pavogtą kriptovaliutą jos savininkams [**TheDAO**].

Nepaisant išvardytų priežasčių, technologijos populiarumas neblėsta. Blokų grandinės technologiją pradeda naudoti didžiosios kompanijos („Microsoft“ bei „IBM“ siūlo blokų grandinę kaip paslaugą [**Zheng2017**]). Taip pat, vis daugiau programuotojų kuria išmaniuosius kontraktus (darbo rašymo metu „Ethereum“ blokų grandinėje yra 25374 verifikuoti išmanieji kontraktai [**EthVerifiedContracts**]).

Blokų grandinė dar nėra brandi, ilgai taikoma technologija. Ji aktyviai bandoma tiek startuolių, tiek korporacijų, vis dar intensyviai tiriama mokslininkų. Technologiją naudojančių programų sėkmė nulems, ar asmenys sutiks naudotis blokų grandine paremtomis programomis.

5. Tapatybės atributų valdymo modelio prototipas

Tapatybės atributų valdymo modelis sudarytas iš 3 dalių: atributų saugyklos kontrakto, paslaugų registro kontrakto bei atributų valdymo programos (žr. 3.2.1 sk.). Siekiant patvirtinti galimą modelio veikimą, „Solidity“ programavimo kalba „Ethereum“ blokų grandinei realizuoti atributų saugyklos bei paslaugų registrų kontraktai.

5.1. Pasirinktos technologijos

Išmaniesiems kontraktams programuoti pasirinkta naudoti „Ethereum“ platformą ir „Solidity“ programavimo kalbą išmaniesiems kontraktams rašyti. Priežastys tam:

- programuotojų bendruomenė. Nuo 2015-ųjų veikianti „Ethereum“ platforma turi nemažą palaikančių asmenų bendruomenę, kuri prisideda prie pačio „Ethereum“ atviro kodo vystymo bei atsako į pradedančiųjų klausimus;
- technologijos branda. Kai išmaniųjų kontraktų programavimas yra vis dar gana nauja sritis, „Ethereum“ technologija, veikianti nuo 2015-ųjų, yra gana gerai dokumentuota, o rekomenduojama kontraktams rašyti programavimo kalba „Solidity“ [**Ethereum**] turi keletą skirtingų kūrimo karkasų programavimo aplinkai paruošti [**SolidityDocumentation**].

Išmanieji kontraktai kurti naudojant „Solidity“ programavimui skirtą „Truffle“ kūrimo karkasą, taikant „ganache-cli“ lokalią blokų grandinę testavimui. Kontraktų funkcijos išbandytos jas kviečiant iš sukurto testinio „React“ puslapio. Kodas pasiekiamas „GitHub“ repozitorijoje `https://github.com/Ubaldas11/blockchain-attribute-management`.

Išmaniosios programėlės maketai kurti su X prototipavimo įrankiu.

5.2. Atributų saugyklos kontraktas

Kontraktas skirtas naudotojų atributų saugojimui bei prieigų suteikimui realizuoti. Kontraktas sudarytas iš kontrakto saugyklos, kurioje saugomos koduotos atributų reikšmės ir jų prieigos, įvykių bei funkcijų. Kontrakto kodas pateikiamas priede nr. 2.

Kontrakte apibrėžti šie įvykiai:

1. *AccessRequested*. Skirtas pranešti, kad paslauga P prašo prieigos prie naudotojo N atributo A. Šio įvykio laukia atributų valdymo programa ir jį gavusi, gali pranešti naudotojui N apie naują paslaugos P užklausą.
2. *AccessChanged*. Skirtas pranešti, kad naudotojas N pakeitė paslaugos P prieigą prie atributo A. Paslaugos, laukiančios šio įvykio, jį gavusios sužino apie galimai suteiktą arba panaikintą prieigą prie atributo A.

Kontrakte apibrėžtos šios viešos funkcijos:

1. *grantAccess*. Skirta naudotojui N suteikti paslaugai P prieigą prie atributo A. Į funkciją paduodamas A identifikatorius, P adresas bei N ir P šifro raktu užkoduota A reikšmė. Funkcijoje keičiama siuntėjo atributo reikšmė - taip užtikrinama, kad naudotojas N negalės pakeisti naudotojo N1 atributo. Funkcija paskelbia *AccessChanged* įvykį;
2. *removeAccess*. Skirta naudotojui N panaikinti paslaugos P prieigą prie atributo A. Į funkciją paduodamas A identifikatorius ir P adresas. Kaip ir *grantAccess* atveju, keičiama siuntėjo atributo prieiga. Funkcija paskelbia *AccessChanged* įvykį;
3. *requestAccess*. Skirta paslaugai P pranešti, kad ji nori prieigos prie naudotojo N atributo A reikšmės. Į funkciją paduodamas A identifikatorius bei N adresas. Funkcija paskelbia *AccessRequested* įvykį;
4. *getAttribute*. Skirta gauti naudotojo N paslaugai P skirtą atributą A. Į funkciją paduodamas A identifikatorius, N adresas bei P adresas. Jei funkciją iškvietė N (t.y., pats naudotojas kreipiasi gauti suteiktą atributą A), grąžinamas siekiamas atributas. Jei kreipiasi paslauga, galimos 3 grąžinamos reikšmės. Jei prieiga naudotojo suteikta - grąžinama A reikšmė. Jei prieiga atmesta - pranešama, kad paslauga neautorizuota. Jei prieiga naudotojo dar nebuvo svarstyta, prašoma pirma kreiptis į *requestAttributeAccess* funkciją.
   Funkcija realizuota kaip vaizdo (angl. *view*) funkcija - ji nekeičia kontrakto būsenos, todėl jos kvietimas yra nemokamas.

Kadangi įvykių paskelbimas yra pigesnis nei kontrakto būsenos keitimas, tai išnaudota paskelbiant apie naują prieigos užklausą (vietoj alternatyvos pakeisti kontrakto būseną ir nurodyti, kad paslauga kreipėsi). Jei paslauga arba atributų valdymo programa realiu laiku „neišgirdo“ įvykio, ji vėliau gali pasiekti visą įvykių sąrašą (angl. *log)*.

5.3. Paslaugų registro kontraktas

Kontraktas skirtas užsiregistravusių paslaugų sąrašui saugoti. Paslaugų siųstuose įrašuose laikomas registracijos kodas bei paslaugos pavadinimas. Kontrakto kodas pateikiamas priede **??**.

Kontraktą sudaro dvi funkcijos:

1. *register*. Skirta paslaugai užsiregistruoti. Į funkciją paduodamas registracijos kodas bei paslaugos pavadinimas, kurie išsaugomi kontrakto būsenoje.
2. *getService*. Skirta atributų valdymo programai išgauti duomenis apie besikreipusią paslaugą. Funkcijai paduodamas paslaugos adresas. Funkcija realizuota kaip vaizdo (angl. *view*) funkcija - ji nekeičia kontrakto būsenos, todėl jos kvietimas yra nemokamas.
   Atributų valdymo programa kreipiasi į šią funkciją tuomet, kai gauta nauja paslaugos P atributo A užklausa. Iš gauto kodo programa gali atskirti, ar ši paslauga nėra apsimetelė (žr. 3.2.3 skyrelį). Jei kodas nesutampa, atributo prieigos užklausą galima ignoruoti. Jei kodas sutampa, gautą pavadinimą galima rodyti atributų valdymo programoje ir pranešti apie prieigos užklausą naudotojui.

6. Kiti blokų grandinės projektai tapatybės valdymo srityje

Tiriant blokų grandinės panaudojimą skaitmeninės tapatybės valdymo srityje, rasta įvairių technologiją bandančių pritaikyti projektų. Šis skyrius skirtas trumpai apžvelgti autoriaus nuomone įdomiausius projektus, susijusius su tapatybe bei blokų grandinės technologija.

6.1. Blockstack

„Blockstack“ projektas yra kompiuterių tinklas, kolektyviai saugantis globalų domenų vardų bei jų viešų raktų registrą. Su šiuo registru, „Blockstack“ įgyvendiną decentralizuotą domenų vardų sistemą, kur kiekvienas norintis gali užregistruoti savo domeną [**BlockstackWhitepaper**]. „OneName“ projektas, naudojantis „Blockstack“ sistemą, siekia sukurti decentralizuotą tapatybę, kurią susikūręs naudotojas galėtų atsisakyti visų kitų turimų socialinių tinklų („Facebook“, „Google“) tapatybių.

6.2. UPort

„Consensys“ įmonės „UPort“ projektas panašus į „OneName“ - jis skirtas naudotojams turėti decentralizuotas skaitmenines tapatybes, registruotas „Ethereum“ tinkle, su kuriomis galėtų atlikti „Ethereum“ transakcijas. Naudotojai su „UPort“ paskyra galėtų prisijungti prie paslaugų bei valdyti savo atributus. Šio darbo rašymo metu „UPort“ turi parengę integracijas paslaugoms ir elektroninę piniginę, skirtą „Ethereum“ raktų ir transakcijų valdymui, tačiau mobilioji programėlė „UPort“ skaitmeninei tapatybei valdyti vis dar ruošiama (angl. *under development*) [**UPort**].

6.3. Tradle

„Tradle“ yra blokų grandine paremtas projektas, skirtas finansinėms institucijoms (bankams, draudimo įmonėms) įgyvendinti „pažink savo klientą“ (angl. *Know Your Customer - KYC*) procesus, suteikiant klientams didesnę asmens duomenų kontrolę. „Tradle“ suteikia klientams programinę įrangą, kuri padės integruoti „Tradle“ platformą su jų įmonėse jau egzistuojančia klientų duomenų saugojimo infrastruktūra. Klientams išreikštinai sutikus, „Tradle“ taip pat suteikia galimybę perduoti asmenų duomenis iš vienos finansinės įmonės į kitą [**Baars2016**].

Rezultatai ir išvados

Darbo rezultatai:

1. Apžvelgti pagrindiniai skaitmeninės tapatybės valdymo būdai internete.
2. Įvardytos esminės naudotojams kylančios problemos tapatybės valdyme: nepakankama asmens duomenų kontrolė, mažėjantis pasitikėjimas paslaugų duomenų valdymu ir didelė priklausomybė nuo tapatybės tiekėjų.
3. Pristatyta blokų grandinės technologija. Nustatyta, kad galimybė kurti decentralizuotus, veikimą atskleidžiančius išmaniuosius kontraktus blokų grandinėje suteikia pagrindo naudoti blokų grandines ir išmaniuosius kontraktus skaitmeninės tapatybės valdymo srityje.
4. Remiantis į naudotoją orientuotos tapatybės (angl. *user-centric identity*) principais, parengti reikalavimai skaitmeninės tapatybės atributų valdymo modeliui, akcentuojantys įvardytas naudotojų problemas.
5. Paruoštas blokų grandine paremtas tapatybės atributų valdymo modelis, įgyvendinantis išskeltus reikalavimus ir suteikiantis asmeniui galimybę kontroliuoti savo asmens duomenis bei autorizuoti paslaugų prieigą prie jų. Modelis įvertintas išskiriant jo privalumus, trūkumus bei galimus pritaikymo barjerus.
6. Sukurtas pristatyto modelio prototipas. „Solidity“ programavimo kalba realizuoti „Ethereum“ blokų grandinei pritaikyti išmanieji kontraktai, įgyvendinantys paslaugų autorizavimo ir atributų saugojimo logiką. Paruošti mobiliosios programėlės, skirtos naudotojui palengvinti sąveiką su blokų grandine, maketai.

Darbo išvados:

Skaitmeninės tapatybės valdymas yra sritis, kurioje naudotojai neturi pakankamai kontrolės. Dideli kiekiai asmens duomenų yra perduodami tarp programų neinformuojant naudotojų ar nutekinami dėl programišių įsilaužimų. Skaitmeninio identiteto centralizavimas tapatybės tiekėjų sistemose sukuria situaciją, kurioje naudotojas tampa visiškai priklausomas nuo tapatybės tiekėjo bei yra priverstas pasitikėti juo.

Blokų grandinės technologija gali padėti išspręsti susidariusias problemas. Decentralizuotai saugant asmens duomenis, panaikinama priklausomybė nuo tapatybės tiekėjo. Viešuose, visiems prieinamuose išmaniuosiuose kontraktuose aprašius paslaugų autorizavimo logiką, naudotojas gali įsitikinti, kad jis geba kontroliuoti paslaugų turimas prieigas prie jo asmens duomenų. Dėl kontrakto kodo nekintamumo, asmuo gali būti tikras, kad laikui bėgant sąlygos nepasikeis ir naudotojo priimti autorizavimo sprendimai išliks.

Aprašytas blokų grandinės taikymas tapatybės valdymo sistemoje turi iššūkių. Decentralizuotas kodo vykdymas yra gana brangus, įmanomos blokų grandinės tinklo daugumos atakos gali atgrasyti potencialius naudotojus, o tapatybės duomenų iškėlimas į blokų grandinę verčia paslaugų tiekėjus kurti papildomą integraciją. Nepaisant to, galimybė grąžinti asmenims jų duomenų kontrolę internete ir atgauti naudotojų pasitikėjimą gali būti pakankamos paskatos taikyti blokų grandinę skaitmeninės tapatybės valdyme.

Sąvokų apibrėžimai

Adresas - blokų grandinės kontekste tai identifikatorius, blokų grandinėje identifikuojantis paskyrą. Paskyra gali būti asmens arba išmaniojo kontrakto.

Atpažinimo duomenys (angl. *credentials*) - tai duomenys, skirti asmens autentifikavimui. Jie gali būti asmens atributai (pvz. biometriniai duomenys, tokie kaip pirštų antspaudai ar balsas) ar sugalvota informacija (pvz. slapyvardis ir slaptažodis). Dažniausiai internete naudojami atpažinimo duomenys yra identifikatoriaus (slapyvardžio ar el. pašto) ir slaptažodžio pora [**Maler2008**].

Atributas - charakteristika, susieta su esybe (pvz. fiziniu asmeniu). Galimi fizinio asmens atributai: gimimo data, vardas, ūgis, pirštų antspaudai [**Camp2004**]. Atributas gali būti laikinas (pvz. adresas) arba nuolatinis (pvz. asmens kodas).

Autentifikavimas - tai procesas, kurio metu patvirtinama sąsaja tarp tapatybės ir jos identifikatoriaus (t.y., įrodoma, kad asmuo iš tikrųjų yra tas, kas sakosi esąs) [**Camp2004**; **Strictest2011**]. Šiam patvirtinimui naudojami atpažinimo duomenys. Autentifikavimo pavyzdys: *tavo pateiktas slapyvardis ir slaptažodis patvirtina, kad tu esi Jonas Jonaitis*.

Autorizavimas - tai procesas, kurio metu leidžiama arba draudžiama asmeniui atlikti konkretų veiksmą, priklausomai nuo jo identifikatoriaus ar atributo [**Camp2004**]. Pavyzdys: *kadangi tau yra daugiau nei 18 metų, tu gali nusipirkti energetinį gėrimą*.

Identifikatorius - tai atributas, kuris vienareikšmiškai susiejamas su jį pateikiančiu asmeniu ir kurį sunku arba neįmanoma pakeisti. Fizinio asmens identifikatoriaus pavyzdys galėtų būti gimimo data (žmogus gali apie ją meluoti, tačiau gimimo datos pakeisti neįmanoma) [**Camp2004**]. Skaitmeninio identifikatoriaus pavyzdys yra naudotojo elektroninio pašto adresas.

Identifikavimas - tai procesas, kurio metu asmuo susiejamas su jo identifikatoriumi [**Camp2004**]. Identifikavimo pavyzdys yra asmens ir jo vardo susiejimas: *tu esi Jonas Jonaitis*.

Paslaugų tiekėjas (angl. *service provider*) - tai betkokia taikomoji programa, kuri suteikia naudotojui tam tikrą paslaugą ar norimą turinį. Galimi paslaugų tiekėjai yra interneto tinklapiai, susirašinėjimo programos ar kitos taikomosios programos, į kurias kreipiasi naudotojas [**Pashalidis2003**; **Samar1999**]. Paslaugų tiekėjas gali turėti vieną ar kelias paslaugas, kurioms reikia tapatybės valdymo funkcijų. Darbe paslaugų tiekėjas dar gali būti vadinamas pasikliaujančiąja šalimi (angl. *relying party*).

Prieigos raktas (angl. *token*) - tai objektas, identifikuojantis skaitmeninę tapatybę [**TokenDefinition**]. Šis raktas būna išduodamas tapatybės tiekėjo ir skirtas identifikuoti naudotoją. Raktas būna prisegtas prie visų autentifikuoto naudotojo užklausų ir leidžia paslaugos tiekėjui žinoti, koks naudotojas kreipiasi.

Skaitmeninė tapatybė - abstrakti fizinės esybės reprezentacija, sudaryta iš aibės esybės nuolatinių ar laikinų atributų, kurie susiejami su fizine esybe [**Glasser2009**; **Camp2004**]. Fizinė esybė gali būti fizinis arba juridinis asmuo. Šiame darbe, jei nenurodyta kitaip, kalbama apie fizinio asmens skaitmeninę tapatybę.

Skaitmeninės tapatybės valdymas (angl. *digital identity management*) - tai veiksmų, skirtų kontroliuoti tapatybę ir su ja susijusius procesus, visuma [**Dabrowski2008**]. Į tai įeina autentifikavimas, autorizavimas, prieigų kontrolė, tapatybės gyvavimo ciklo valdymas bei saugus tapatybės

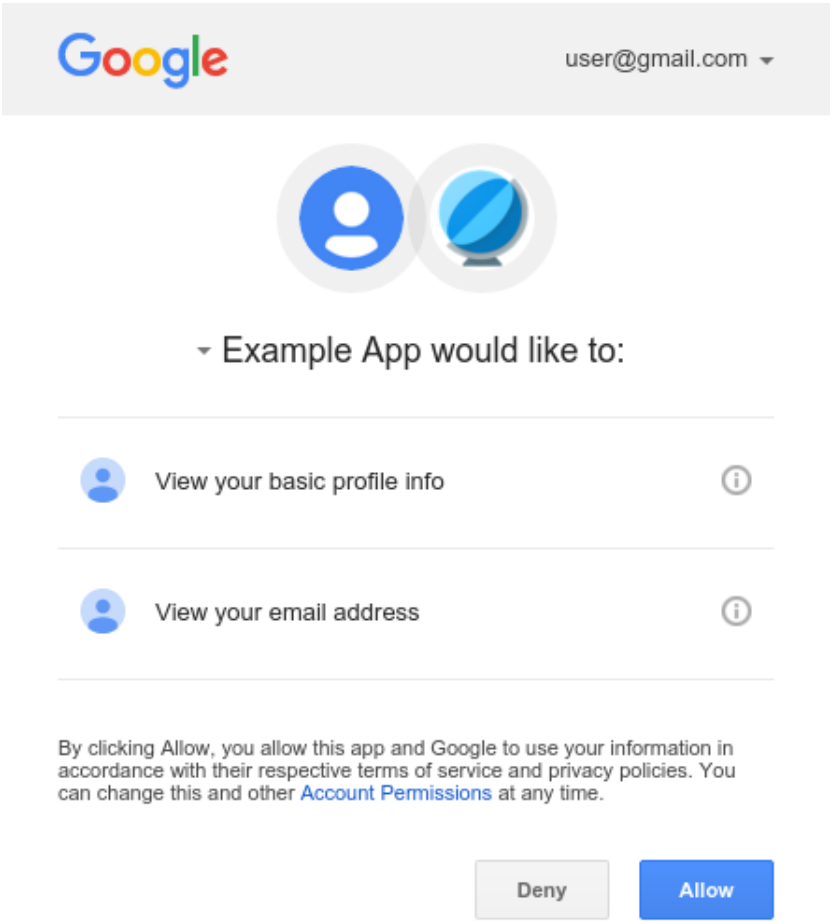
atributų perdavimas trečiosioms šalims [**Cao2010**].

Tapatybės tiekėjas (angl. *identity provider*) - servisas ar taikomoji programa, skirta koordinuoti su tapatybe susijusius duomenis tarp naudotojų, jų naršyklių bei paslaugų tiekėjų [**Strictest2011**]. Pagrindinės tapatybės tiekėjo funkcijos: infrastruktūros naudotojų tapatybės duomenims apdoroti sukūrimas ir užklausų iš paslaugų tiekėjų bei naudotojų apdorojimas [**Cao2010**].

Vienkartinis prisijungimas (angl. *single sign on*) - tai mechanizmas, kuris naudoja vieną tapatybės patvirtinimo veiksmą suteikti naudotojui prieigą prie susijusių, bet nepriklausomų taikomųjų programų ar interneto svetainių, neprašant naudototojo iš naujo prisijungti prie kiekvienos iš jų konkrečios sesijos metu [**Radha2012**].

Priedas nr. 1

OAuth prašymas naudotojui autorizuoti paslaugą



12 pav. Naudotojo sutikimas leisti Example App programai pasiekti jo Google paskyros duomenis

Priedas nr. 2

Atributų saugojimo ir autorizavimo kontraktas

```
pragma solidity ^0.4.23;

contract UserAttributeStore {
    struct ServiceAttribute {
        bool accessGranted;
        string value;
    }
    mapping (address => mapping(uint => mapping(address => ServiceAttribute)))
     private attributes;

    event AccessRequested (
        address userAddress,
        address serviceAddress,
        uint attributeId
    );
    event AccessChanged (
        address userAddress,
        address serviceAddress,
        uint attributeId,
        bool status
    );

    function grantAccess(uint attributeId, address serviceAddress, string
    value) public {
        changeAccess(attributeId, serviceAddress, true, value);
    }

    function removeAccess(uint attributeId, address serviceAddress) public {
        changeAccess(attributeId, serviceAddress, false, "X");
    }

    function changeAccess(uint attributeId, address serviceAddress, bool
    granted, string value) private {
        attributes[msg.sender][attributeId][serviceAddress].accessGranted =
    granted;
        attributes[msg.sender][attributeId][serviceAddress].value = value;
        emit AccessChanged(msg.sender, serviceAddress, attributeId, granted);
    }

    function requestAccess(uint attributeId, address userAddress) public {
        emit AccessRequested(userAddress, msg.sender, attributeId);
    }

    function getAttribute(uint attributeId, address userAddress, address
    serviceAddress)
```

```
41          public
42          view
43          returns (string)
44      {
45          if (msg.sender == userAddress) {
46              return attributes[msg.sender][attributeId][serviceAddress].value;
47          }
48          //to avoid additional 'initialized' property, length check used
49          if (bytes(attributes[userAddress][attributeId][serviceAddress].value).
    length == 0) {
50              return "Please request access first";
51          } else {
52              if (attributes[userAddress][attributeId][serviceAddress].
    accessGranted) {
53                  return attributes[userAddress][attributeId][serviceAddress].
    value;
54              } else {
55                  return "Access denied";
56              }
57          }
58      }
59  }
```

Priedas nr. 3

Paslaugų registro kontraktas

```
pragma solidity ^0.4.23;

contract ServiceRegister {

    struct Service {
        string registrationCode;
        string name;
    }
    mapping (address => Service) private services;

    function register(string registrationCode, string name) public {
        services[msg.sender].registrationCode = registrationCode;
        services[msg.sender].name = name;
    }

    function getService(address serviceAddress) public view returns (string, string) {
        return (services[serviceAddress].registrationCode, services[serviceAddress].name);
    }
}
```