VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ KATEDRA

Savarankiška suvereni tapatybė

Self Sovereign Identity

Magistro baigiamasis darbas

Atliko: Vytautas Saulis
Darbo vadovas: asist. dr. Vytautas Valaitis
Darbo recenzentas: doc. dr. Vytautas Čyras

Vilnius – 2023

TURINYS

ĮVADAS

Savarankiška suvereni tapatybė (SST) – tai skaitmeninės tapatybės valdymo būdas, pagal kurį asmenys kontroliuoja savo asmeninę informaciją ir tai, kaip ja dalijamasi su kitais. SST leidžia asmenims valdyti savo tapatybes, nepriklausomai nuo centralizuotų institucijų, trečiųjų šalių ir tarpininkų [CA16].

Duomenų valdymas ir apsauga tapo itin svarbiu skaitmeninės tapatybės įgyvendinimo reikalavimu. Tradicinės tapatybės sistemos, kurioms būdinga centralizacija ir priklausomybė nuo trečiųjų šalių, dažnai nesuteikia asmenims pilnos jų skaitmeninės tapatybės kontrolės ir nuosavybės [CA16]. Savarankiškos suverenios tapatybės atsiradimas susilaukė didelio dėmesio kaip perspektyvus sprendimas šiems iššūkiams spręsti, nes asmenims suteikiama tapatybės valdymo kontrolė [LA20].

Pastaraisiais metais išaugo akademinių tyrimų skaičius, o tai rodo šios srities tyrimų svarbą. Pavyzdžiui, Möller ir Zeng aptaria savarankiškos suverenios tapatybės, kaip naujos skaitmeninės tapatybės valdymo paradigmos, iškilimą, analizuojant jos sąvokas, technologijas ir realius taikymus [LA20]. Be to, Smithas ir Johnsonas (2021) nagrinėja savarankiškos suverenios tapatybės poveikį privatumo ir duomenų apsaugos reglamentams, pabrėžiant, kad teisinės sistemos turi prisitaikyti prie šio besiformuojančio skaitmeninės tapatybės modelio.

Pagrindinės savarankiškos suverenios tapatybės dalys yra išskirstytos apskaitos technologija (DLT), patvirtinami įgaliojimai (VC) ir decentralizuoti identifikatoriai (DID).

DLT, kurios pavyzdys yra blokų grandinės technologija, siūlo decentralizuotą ir skaidrią realizaciją, užtikrinančią duomenų vientisumą, nekintamumą ir saugumą. Pasinaudojant DLT būdingomis savybėmis, savarankiška suvereni tapatybė tampa įgyvendinama, nes leidžia asmenims užtikrinti savo tapatybės kontrolę, nesikliaujant centralizuotomis institucijomis.

VC atlieka pagrindinį vaidmenį SST realizacijoje, nes suteikia galimybę asmenims saugiai valdyti savo skaitmeninės tapatybės duomenis. Šie įgaliojimai yra skaitmeninės asmeninių savybių, kvalifikacijos ar priklausomybės reprezentacijos ir gali būti kriptografiškai pasirašyti atitinkamų leidėjų. Jie leidžia asmenims pasirinktinai atskleisti tik konkrečioms operacijoms ar sąveikai būtiną asmeninę informaciją, taip padidinant privatumą ir sumažinant duomenų saugumo pažeidimų riziką.

DID, papildantys patvirtinamus įgaliojimus, yra visuotinai unikalūs asmenų, organizacijų ar daiktų identifikatoriai. DID kriptografiškai generuojami ir susiejami su atitinkamais patvirtinamais įgaliojimais, todėl asmenys gali patvirtinti ir kontroliuoti savo skaitmeninę tapatybę įvairiose platformose ir paslaugose. DID suteikia standartizuotą, sąveikų decentralizuotų identifikatorių kūrimo, valdymo ir tikrinimo metodą, taip sudarant sąlygas sklandžiai sąveikai ir su tapatybe susijusios informacijos perkeliamumui.

Šio darbo tikslas – sukurti savarankiškos suverenios tapatybės sistemos prototipą ir patikrinti jo veikimą.

Magistro darbe yra išsikelti šie uždaviniai:

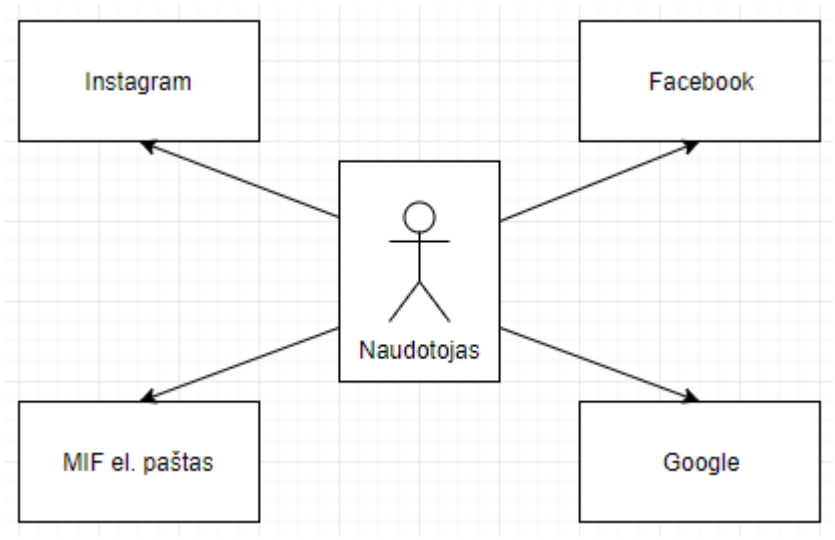
1. Identifikuoti SST saugumo grėsmes ir problemas bei pateikti grėsmių ir problemų sprendimus.
2. Sukurti SST sistemos architektūrą pagal W3C decentralizuotų identifikatorių specifikaciją.
3. Sukurti SST sistemos prototipą pagal sukurtą SST sistemos architektūrą.
4. Remiantis SST technologijos bendraautoriaus Kristoferio Aleno pasiūlytais SST technologijos principais [RSLA19, CA16], patikrinti sukurtą SST sistemos prototipą.

1. Skaitmeninė tapatybė

Skaitmeninė tapatybė – duomenys apie subjektą, kuriuos kompiuterinės sistemos naudoja identifikuoti esybei. Esybė gali būti fizinis ir juridinis asmuo, programinė įranga ar įrenginys. Elektroninė informacija susieta su asmeniu tam tikroje tapatybių sistemoje vadinama skaitmenine tapatybe [DB16]. Tapatybių sistemos gali būti naudojamos autentifikavimui ir autorizavimui. Autentifikavimas yra procesas skirtas patvirtinti naudotojo tapatybę. Trys pagrindiniai metodai naudojant autentifikuoti žmogų:

- Žinios (slaptažodis, PIN kodas).
- Prietaisai (kodų generatorius, išmanioji kortelė).
- Biometrija (pirštų antspaudas, veido forma).

Procesas, kurio metu nustatoma ką asmuo gali daryti po to kai jis yra autentifikuojamas, vadinama autorizavimu. Remiantis pasauline interneto statistika[1], 2019 m. kovo 31 d. buvo daugiau kaip keturi milijardai interneto naudotojų, kas sudaro daugiau negu 50 % visų pasaulio gyventojų. Šalia skaitmeninės tapatybės, kurią sukuria interneto tiekėjas, naudotojai dažniausiai turi daugiau skaitmeninių tapatybių socialiniuose tinkluose, bankuose, mobiliosiose programėlėse ar kituose interneto puslapiuose (1 pav.).

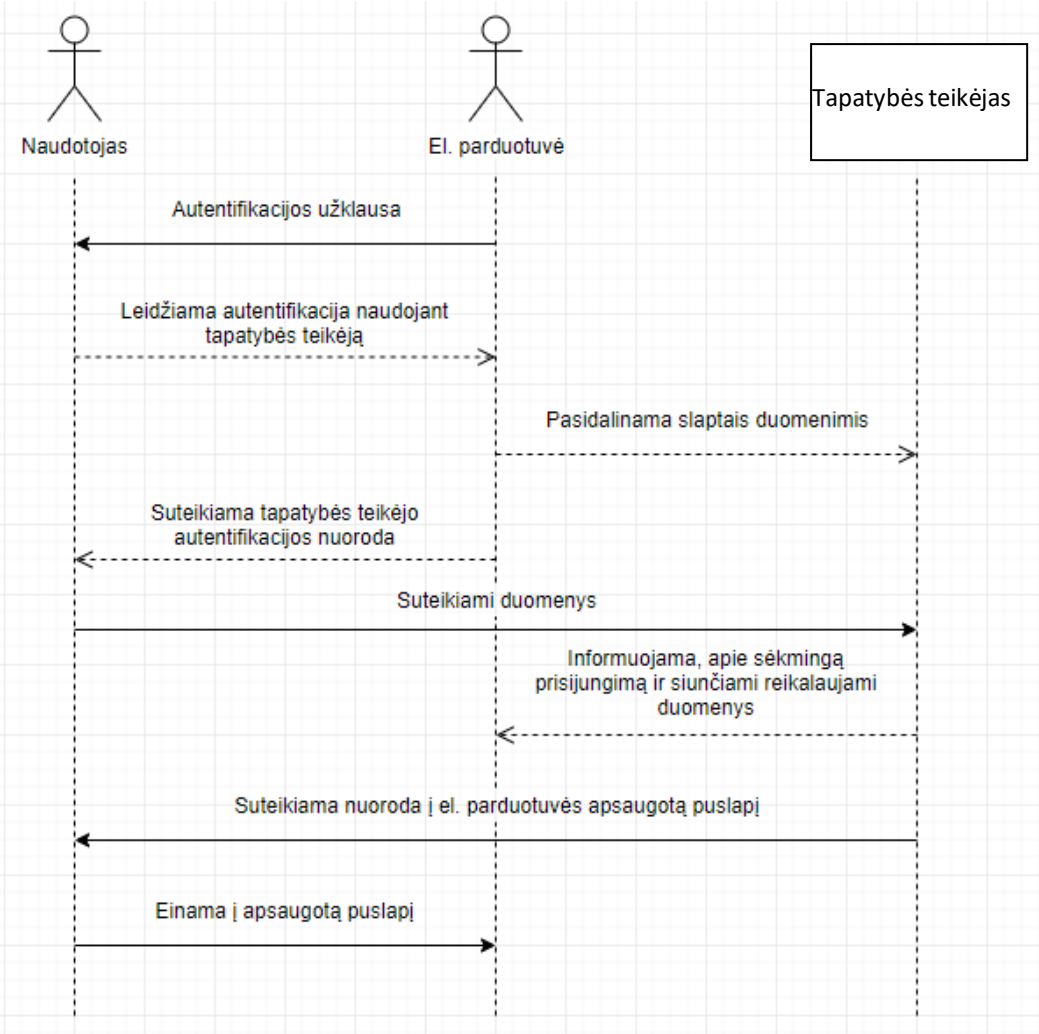


1 pav. **Keletas naudotojo skaitmeninių tapatybių [DB16]**

Skaitmeninių tapatybių valdymas tampa vis svarbesnis dėl augančio internetinių puslapių ir mobiliųjų programėlių, reikalaujančių patvirtinti savo tapatybę, skaičiaus. Nors keletas organizacijų, tokių kaip „OpenID“, suteikia galimybę interneto naudotojams lengviau valdyti savo skaitmeninę tapatybę, naudojantis vienkartinio prisijungimo paslauga (angl. single sign-on, toliau

[1] https://www.internetworldstats.com/stats.htm

– SSO), tačiau tai nėra sprendimas suteikiantis galimybę naudotojams valdyti ir saugoti savo skaitmeninę tapatybę patiems. Prisijungimo žingsniai naudojant SSO pavaizduoti 3 pav. Organizacijos siūlančios skaitmeninės tapatybės valdymo sistemas turi galimybę saugoti duomenis, kas su kuo bendradarbiauja ir kas kur yra užsiregistravęs, tai sukelia rimtas privatumo problemas. Organizacijos siūlančios platų paslaugų spektrą turi galimybę lengvai susieti duomenis, taip galėdamos naudotojus skirstyti į tikslines auditorijas.



2 pav. **Prisijungimas naudojant vienkartinio prisijungimo paslaugą [DB16]**

1997 metais *Microsoft* sukūrė „Miscrosoft Passport“ sistemą, kuri suteikė galimybę naudoti tą pačią skaitmeninę tapatybę kitoje programinėje įrangoje. Šis sprendimas vėliau pavadintas susieta tapatybe (angl. federated identity). Nepaisant *Microsoft* reputacijos, duomenys buvo saugomi *Microsoft* organizacijos rėmuose, kas reiškia, kad šis sprendimas niekuo nesiskiria nuo kitų centralizuotai asmens duomenis saugančių tapatybių sistemų. Vienas iš pagrindinių tokių sistemų trūkumas, kad naudotojas yra priklausomas nuo organizacijos ir atsiradus klaidoms organizacijos

sistemoje arba organizacijai, nutraukus veiklą, kartu su ja būtų prarasta irnaudotojo skaitmeninė tapatybė.

1.1. Savarankiška suvereni tapatybė

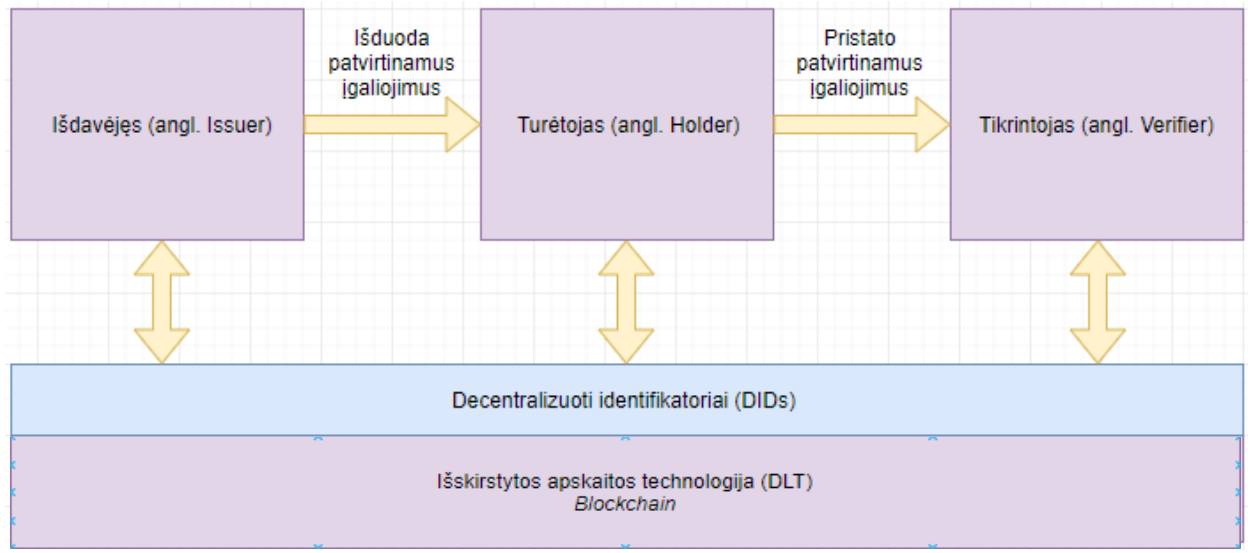
Skaitmeninė tapatybė bėgant laikui tobulėjo ir keitėsi, o dėl nepatenkintų rinkos poreikių atsirado nauja skaitmeninės tapatybės versija – savarankiškai suvereni tapatybė (SST). Pati sąvoka nėra nauja, tačiau jos apibrėžimas ir svarba yra aktuali iki šių dienų. Vienas pirmųjų paminėjęs šią technologiją ir pasiūlęs šį apibrėžimą savo tinklaraštyje buvo Kristoferis Alenas, decentralizuotų identifikatorių (angl. decentralized identifiers, toliau – DID) ir išskirstytos apskaitos technologijos (angl. distributed ledger technology, toliau – DLT) bendraautorius [RSLA19, CA16]. Tame įraše jis apibūdino skaitmeninių tapatybių evoliuciją ir pasiūlė šį terminą: „Savarankiška suvereni tapatybė yra kitas žingsnis po centrinės naudotojo tapatybės ir tai reiškia, kad ji prasideda toje pačioje vietoje – vartotojas turi būti tapatybės administravimo centre“ [CA16]. Joe Andrieu savarankišką suverenią tapatybę apibrėžė kaip sistemą, kuri leidžia selektyviai pasirinkti savo identifikavimo priemones, kurios kontroliuoja sąveiką formaliose ir neformaliose situacijose visame pasaulyje [JA16]. Magistro darbo autorius šį išsireiškimą interpretuoja, kad „...kitas žingsnis...“ yra minimas, norint pabrėžti SST padidėjusį suverenitetą, lyginant su centrine naudotojo tapatybe, kurios duomenys yra saugomi centralizuotoje duomenų saugykloje. Jis išskyrė10 pagrindinių SST principų:

- Egzistavimas (angl. existence) – esybės negali būti tik skaitmeninės (pvz. fiziniai ir juridiniai asmenys, elektroniai prietaisai).
- Valdymas (angl. control) – esybės turi visada valdyti savo tapatybes.
- Prieiga (angl. access) – esybės turi tiesioginę prieigą prie savo tapatybės ir susijusių duomenų. Visi duomenys turi būti pasiekiami ir matomi be trečiųjų šalių.
- Skaidrumas (angl. transparency) – sistemos logika bei funkcionalumas turi būti valdomas ir atnaujinamas skaidriai.
- Patvarumas (angl. persistence) – tapatybės turi būti ilgaamžės.
- Perkeliamumas (angl portability) – visi saugomi duomenys apie tapatybes turi būti perkeliami. Tapatybė negali priklausyti trečiosioms šalims.
- Sąveika (angl. interoperability) – tapatybės turėtų būti naudojamos taip plačiai kaip įmanoma.
- Sutikimas (angl. consent) – esybės turi duoti sutikimą naudojant jų tapatybes ir dalinantis susijusia informacija.
- Sumažinimas (angl. minimization) – trečiosios šalys gali pasiekti tik tikslui įgyvendinti reikalingus esybės duomenis.

- Apsauga (angl. protection) – esybių teisė būti apsaugotomis. Jeigu kyla konfliktas tarp interneto poreikių ir esybių teisių, pirmenybė turi būti teikiama esybių teisėms.

Kristoferio Aleno tinklaraščio įrašo pabaigoje teigiama, kad aukščiau paminėti 10 principų turėtų būti laikomi kaip ateities diskusijų pradžios taškas. 2017 m. pabaigoje kartu su Šenonu Apelklainu Kristoferis Alenas sukūrė kitą savarankiškos suverenios tapatybės apibrėžimą: „Savarankiškai suverenios tapatybės centre yra žmogus, ji yra nepriklausoma nuo bet kokios korporacijos, organizacijos ar valstybės“.

SST technologija išskiria reikalavimus ir siekiamybes tapatybių teikėjams, kurie turėtų juos įgyvendinti, kuriant naujas tapatybių valdymo sistemas. Savarankiškos suverenios tapatybės modelis pavaizduotas 3 pav.



3 pav. **Savarankiškai suverenios tapatybės modelis [CA16, SLC19]**

2. Išskirstytos apskaitos technologija

Šiame skyriuje nagrinėsime išskirstytos apskaitos technologijos (IAT) taikymą tapatybės valdyme. Išskirstytoji apskaitos technologija, dažnai vadinama blokų grandinės technologija, yra naujas ir inovatyvus informacijos saugojimo, patvirtinimo ir paskirstymo būdas, kuris gali turėti didelę įtaką įvairiose srityse. Šio skyriaus tikslas yra apžvelgti IAT principus, privalumus ir potencialą tapatybės valdyme, su ypatingu dėmesiu skiriant savarankiškai suvereninei tapatybei.

Plačiausiai žinoma DLT yra *Blockchain*, kur transakcijų apsauga, integravimas ir validavimas įgyvendinami, jungiant duomenų blokus į grandinę. DLT tapo viena iš kertinių savarankiškai suverenios tapatybės dalių, o *Blockchain* dažnai yra tapatinama su savarankiškai suverenia tapatybe, tačiau svarbu suprasti, kad tai yra skirtingos technologijos, kurios gali būti naudojamos nepriklausomai viena nuo kitos. Savarankiškai suverenios tapatybės technologijoje, nėra nurodyta

būtinybė naudoti DLT, todėl gali būti naudojamas kitas savarankiškos suverenios tapatybės principus įgyvendinantis duomenų saugojimo modelis. Kai tik duomenys yra įterpiami į *Blockchain* blokų grandinę, jie negali būti ištrinti ar pakeisti.

3. Patvirtinami įgaliojimai

Įgaliojimai (angl. credentials) yra mūsų kasdienybės dalis. Vairuotojo pažymėjimai yra naudojami patvirtinti, kad turime teisę vairuoti motorinę transporto priemonę, universiteto baigimo pažymėjimai – patvirtinti mūsų išsilavinimo lygį, valstybės išduoti pasai – identifikuoti asmenį, norintį keliauti į užsienio šalis. Šie įgaliojimai mums reikalingi, kai yra naudojami fiziniame pasaulyje, tačiau jų naudojimas virtualioje erdvėje vis dar yra kvestionuojamas. Problematiška pateikti programinei įrangai suprantamą informaciją, kaip išsilavinimas, sveikatos ir finansiniai duomenys bei kiti trečiųjų šalių patvirtinti duomenys. Sunkumai naudotis įgaliojimais virtualioje erdvėje mažina galimybę gauti iš jų tokią pačią naudą virtualioje erdvėje kaip ir fiziniame pasaulyje. W3C sukūrė specifikaciją, kuri nurodo kaip kriptografiškai saugiai, privačiai ir programinei įrangai suprantamai naudotis įgaliojimais virtualioje erdvėje [SLC19]. Įgaliojimai gali būti informacija susijusi su:

- Įgaliojimą turinčiu subjektu (pvz., nuotrauka, vardas, identifikacinis numeris).
- Įgaliojimus išduodančia institucija (pvz., miesto valdžia, nacionalinė agentūra).
- Įgaliojimo tipu (pvz., valstybės pasas, vairuotojo pažymėjimas, sveikatos apsaugos kortelė).
- Institucijų patvirtintais subjekto požymiais (pvz., pilietybė, gimimo data, vairuotojo kategorija).

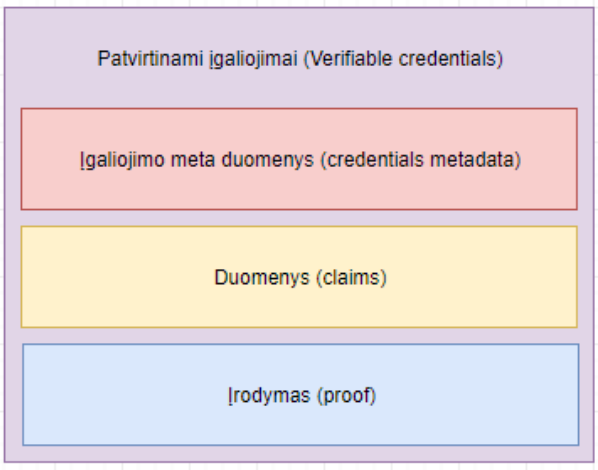
Virtualioje erdvėje patvirtinami įgaliojimai gali vaizduoti tą pačią informaciją kaip ir fiziniame pasaulyje naudojami įgaliojimai. Tokių technologijų, kaip skaitmeninių parašų, pagalba patvirtinami įgaliojimai yra atsparūs apgaulėms ir patikimesni nei jų naudojami atitikmenys fiziniai pasaulyje. Patvirtinamų įgaliojimų turėtojai gali sugeneruoti patvirtinamus pavaizdavimus ir dalintis jomis su tikrintojais, kad įrodytų jų tikrumą. Tiek patvirtinami įgaliojimai, tiek patvirtinami pavaizdavimai gali būti perduodami daug greičiau nei jų fiziniai atitikmenys. Lentelėje nr. 1 yra pateiktos visos W3C specifikacijoje minimos rolės, kurios yra naudojamos patvirtinamų įgaliojimų modeliuose bei yra abstrakčios, todėl gali būti naudojamos skirtingai.

1 lentelė. **Patvirtinamų įgaliojimų rolės [SLC19]**

Rolė	Apibūdinimas
Turėtojas (angl. holder)	Esybės rolė gali turėti vieną arba daugiau patvirtinamų įgaliojimų ir jiems generuoti patvirtinamus pavaizdavimus. Pvz., studentas, darbuotojas, pirkėjas.
Išdavėjas (angl. issuer)	Esybės rolė tvirtina duomenis apie vieną arba keletą subjektų, iš tų duomenų jiems kuria patvirtinamus įgaliojimus ir perduoda juos turėtojui. Pvz., korporacijos, ne pelno siekiančios organizacijos, prekybos asociacijos, vyriausybės irkiti asmenys.
Subjektas (angl. subject)	Esybės rolė apie kurią duomenys yra kuriami. Dažniausiai patvirtinamų įgaliojimų turėtojas yra ir subjektas, bet išskirtiniais atvejais gali ir nebūti, kaip pavyzdžiui tėvai gali turėti savo vaikų patvirtinamus įgaliojimus arba gyvūnų savininkai gali turėti savo gyvūnų patvirtinamus įgaliojimus. Pvz., žmonės, gyvūnai, daiktai.
Tikrintojas (angl. verifier)	Esybės rolė gauna vieną arba keletą patvirtinamų įgaliojimų, pasirinktinai patvirtinamuose pavaizdavimuose, ir juos apdoroja. Pvz., darbdaviai, apsaugos darbuotojai, internetiniai puslapiai.
Patvirtinamų duomenų registras (angl. verifiable data registry)	Sistemos rolė tarpininkauja identifikatorių, viešų raktų, patvirtinamų įgaliojimų schemų ir išdavėjo viešų raktų kūrime ir tikrinime. Pvz., patikimos duomenų bazės, decentralizuotos duomenų bazės, vyriausybės ID duomenų bazės ir išskirstytos apskaitos. Dažniausiai daugiau nei vienas patvirtinamų duomenų registras yra naudojamas ekosistemoje.

3.1. Įgaliojimai

Įgaliojimai yra duomenų rinkinys sukurtas pačios esybės. Taip pat, įgaliojimai gali turėti ir identifikatorių bei meta duomenis skirtus apibūdinti įgaliojimo laukus (pvz. išdavėją, galiojimo pabaigos datą ir laiką, viešą raktą naudojamą verifikavimo tikslams, raktų atgavimo algoritmą ir t. t.). Meta duomenys gali būti pasirašyti išdavėjo. Patvirtinami įgaliojimai yra klastotėms atsparių duomenų rinkinys ir meta duomenys, kuriuos galima kriptografiškai patvirtinti, kas juos išdavė. Patvirtintų įgaliojimų duomenų modelis pavaizduotas 4 pav.



4 pav. **Patvirtinamų įgaliojimų duomenų modelis [SLC19]**

Patvirtinamus įgaliojimus sudaro:

- Kontekstas (angl. context).
- Identifikatorius (angl. identifier).
- Tipas (angl. type).
- Įgaliojimo subjektas (angl. credential subject).
- Įrodymas (angl. proof).

3.1.1. Kontekstas

Patvirtinami įgaliojimai turi daug atributų ir reikšmių, kurie yra identifikuojami pagal URI, tačiau jie gali būti ilgi ir žmogui sunkiai įsimenami. Tokiais atvejais lengvai įsimenami trumpiniai gali būti labai naudingi. W3C specifikacija naudoja *@context* lauką, kad ilgus URI pakeistų jų trumpiniais. *@context* lauko ištrauka yra pavaizduota 5 pav. Šis laukas turi tenkinti šias sąlygas:

- @context turi būti surikiuotas sąrašas, kur pirmas sąrašo elementas yra URI su reikšme: https://www.w3.org/2018/credentials/v1.
- Kiti sąrašo elementai turi nurodyti kontekstinę informaciją.
- Rekomenduojama, kad kiekvienas URI esantis @context lauko reikšmėje, būtų programinei įrangai suprantama informacija apie kontekstą.

```
{
    "@context": [
        "https://www.w3.org/2018/credentials/v1",
        "https://www.w3.org/2018/credentials/examples/v1"
    ]
}
```

5 pav. **Konteksto lauko pavyzdys [SLC19]**

3.1.2. Identifikatorius

Pateikiant informaciją apie žmogų, produktą ar organizaciją, dažnai naudinga naudoti identifikatorius, kad kiti suprastų apie kokį subjektą yra kalbama. W3C specifikacija, identifikatoriams apibrėžia *id* lauką, kuris yra aiški nuoroda į žmogų, produktą ar organizaciją ir suteikia galimybę saugoti informaciją patvirtinamuose įgaliojimuose. *Id* lauko ištrauka yra pavaizduota 6 pav. *Id* laukas turi tenkinti nurodytas sąlygas:

- *Id* laukas turi būti unikalus.
- *Id* lauko reikšmė turi atitikti URI formatą.

```
{
    "id": "did:example:123456789abcdefghi"
}
```

6 pav. **Identifikatoriaus lauko pavyzdys [SLC19]**

3.1.3. Tipas

Programinės įrangos sistemos apdorojančios W3C specifikacijose nurodytus objektus, naudoja *type* lauko reikšmę, kad nustatytų ar yra naudojami patvirtinami įgaliojimai. *Type* lauko ištrauka yra pavaizduota 7 pav. Patvirtinami įgaliojimai turi turėti *type* lauką, o laukas turi tenkinti nurodytas sąlygas:

- *Type* lauko reikšmė turi turėti vieną arba daugiau URI reikšmių.
- Nurodžius daugiau negu vieną URI, URIs sąrašas turi būti interpretuojamas kaip nerikiuotas.
- Rekomenduojama, kad kiekvienas URI, nurodytas *type* lauke, nukreiptų į programinei įrangai suprantamą informaciją JSON-LD formatu.

```
{
    "type": [
        "VerifiableCredential",
        "UniversityDegreeCredential"
    ]
}
```

7 pav. **Tipo lauko pavyzdys [SLC19]**

3.1.4. Įgaliojimo subjektas

Patvirtinami įgaliojimai saugo duomenis apie subjektus. W3C specifikacija apibrėžia *credentialSubjekt* lauką, kad pateiktų duomenis apie subjektus. *CredentialSubjekt* lauko ištrauka yra pavaizduota 8 pav. Patvirtinami įgaliojimai turi turėti *credentialSubject* lauką, o pats laukas turi tenkinti nurodytas sąlygas:

- *CredentialSubject* lauko reikšmė yra apibrėžiama, kaip objektų rinkinys, turintis vieną arba daugiau laukų, kurie apibūdina patvirtinamo įgaliojimo subjektą.
- Kiekvienas objektas nurodytas *credentialSubject* lauko reikšmėje, gali turėti *id* lauką.

```
{
    "credentialSubject": {
        "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
        "degree": {
            "type": "BachelorDegree",
            "name": "<span lang='lt'>Vilniaus Universiteto bakalauras</span>"
        }
    }
}
```

8 pav. **Įgaliojimo subjekto lauko pavyzdys [SLC19]**

3.1.5. Įrodymas

Bent vienas įrodymo mechanizmas ir informacija apie jį yra būtina norint suteikti įrodymui prasmę. Norint, kad įgaliojimai būtų patvirtinti, dokumentas turi turėti *proof* lauko reikšmę. Skirtingos technologijos taikomos pasirašinėjant patvirtinamus įgaliojimus, todėl šio lauko reikšmė gali skirtis. *Proof* laukas yra pavaizduotas 9 pav.

```
    {
        "proof": {
            "type": "LinkedDataSignature2015",
            "created": "2016-02-08T16:02:20Z",
            "creator": "did:example:8uQhQMGzWxR8vw5P3UWH1ja#keys-1",
            "signatureValue": "QNB13Y7Q9...1tzjn4w=="
        }
    }
```

9 pav. **Įrodymo lauko pavyzdys [SLC19]**

4. Decentralizuoti identifikatoriai

Decentralizuotas identifikatorius (DID) yra naujas patikrinamo identifikatoriaus tipas naudojamas savarankiškai suverenios tapatybės technologijos įgyvendinimui [RSLA19]. DID yra kontroliuojami DID subjekto ir yra nepriklausomi nuo centralizuotų duomenų bazių, tapatybės teikėjų ar sertifikato institucijų. DID yra URL, kurie yra vienareikšmiškai susieti su DID subjektu ir skirti patikimam komunikavimui su juo. Kiekvienas DID turi su juo susietus DID dokumentus, kurie nurodo, kaip naudoti su tuo dokumentu susietą DID. Kiekvienas DID dokumentas, gali turėti 3 dalykus:

- Įrodymų tikslus (angl. *proof purposes*).
- Verifikavimo metodus (angl. *verification methods*).
- Paslaugų nuorodas (angl. *services endpoints*).

Įrodymų tikslai yra kombinuoti su verifikavimo metodais, kad sukurtų mechanizmą, gebantį patvirtinti ar pasirašyti dalykus. Pavyzdžiui, DID dokumentas gali įrodyti, kad tam tikras verifikavimo būdas, kaip kriptografijos viešas raktas ar pseudonominis biometrinis protokolas, gali

būti naudojamas įrodyti, kad patvirtinimas ar parašas buvo sukurtas tam tikram autentifikavimo tikslui. Paslaugų nuorodos suteikia patikimą komunikaciją su DID subjektu. DID architektūros principai pateikti lentelėje nr. 2.

Augantis decentralizuotų identifikatorių poreikis suformavo du pagrindinis reikalavimus, kuriuos turi turėti naujo tipo URL, kurie toliau funkcionuotų internetinėje architektūroje ir turėtų kelis papildomus reikalavimus, kurių HTTP naudojami URL neturi:

- Naujo tipo URL turi nepriklausyti nuo centralizuotų institucijų norint užregistruoti, atnaujinti ir deaktyvuoti susijusius duomenis. Didžioji dalis URL yra priklausomos nuo DNS vardų ar IP adresų, kurie taip pat yra priklausomi nuo centralizuotų institucijų. DID turi būti kuriami ir valdomi nepriklausomai nuo visų institucijų.
- URL, kurio valdymas ir susiję meta duomenys, įskaitant viešuosius raktus, gali būtų patikrinti kriptografiškai. Autentifikavimas naudojant DID ir DID dokumentus naudoja viešo ir privataus rakto kriptografiją.

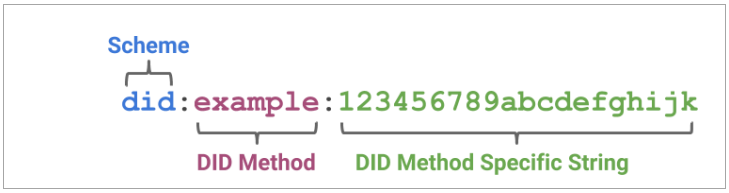
2 lentelė. **DID architektūros principai [RSLA19]**

Tikslas	**Apibūdinimas**
Decentralizavimas (angl.decentralization)	DID architektūra turi eliminuoti centralizuotų institucijų reikiamybę, įskaitant globaliai unikalių identifikatorių, viešųverifikavimo raktų, paslaugų nuorodų ir kitų meta duomenųregistravimo.
Savarankiškas suverenumas (angl. self-sovereignty)	DID architektūra turi užtikrinti esybėms tiek žmogiškoms, tiek nežmogiškoms, jų skaitmeninių identifikatorių kontrolę,be būtinybės pasitikėti išorinėmis institucijomis.
Privatumas (angl. privacy)	DID architektūra turi užtikrinti galimybę kontroliuoti esybėms jų informacijos privatumą, įskaitant minimalų, pasirinktiną ar pilnąjų privačios informacijos atskleidimą.
Saugumas (angl. security)	DID architektūra turi užtikrinti pakankamą saugumą, kad nuo to priklausančios šalys galėtų priklausyti nuo DID dokumentų, užtikrinant reikiamą patikimumo lygį.
Pagrįstumas (angl. proof-based)	DID architektūra turėtų sudaryti sąlygas DID subjektui pateikti kriptografinį autentiškumo patvirtinimo ir patvirtinimo teisių įrodymą.
Atrandamumas (angl. discoverability)	DID architektūra turi suteikti galimybę esybėms atrasti kitųesybių DID, su tikslu daugiau sužinoti apie jas ar su jomis komunikuoti.
Integralumas (angl. interoperability)	DID architektūra turi naudoti standartus, kad DID infrastruktūra galėtų pasinaudoti esamais įrankiais ir programinės įrangos bibliotekomis.
Perkeliamumas (angl. portability)	DID architektūra turi būti nepriklausoma nuo sistemos ir tinklo. Esybėms turi galėti naudoti savo skaitmeninius identifikatorius bet kurioje sistemoje, kuri palaiko DID ir DID metodus.
Paprastumas (angl. simplicity)	Siekiant užtikrinti šiuos projektavimo principus, DID architektūra turėtų būti kiek įmanoma paprasta.
Plečiamumas (angl. extensibility)	DID architektūra turėtų būti kiek įmanoma plečiama, jei tai nekenkia integralumo, perkeliamumo ar paprastumo principams.

4.1. DID

Globaliai unikalaus decentralizuoto identifikatoriaus technologija nėra nauja. Universaliai unikalūs identifikatoriai (UUID) pirmą kartą buvo sukurti 1980 m. ir vėliau tapo viena pagrindinių Viešos programinės įrangos fondo išskirstytos skaičiavimo aplinkos sistemos dalių [RSLA19]. UUIDs pasiekė globalų unikalumą ir nepriklausymą nuo centralizuotų institucijų naudodama algoritmą, kuris sugeneruoja 128 bitų reikšmes, kurių kolizijos tikimybė yra labai maža.

DID technologija yra panaši į UUID, tik DID formatas yra toks pat kaip URL. Jis gali būti suprantamas ir nukreipiamas į standartinį šaltinį apibūdinantį DID subjektą, tačiau priešingai nei URL, DID dokumentas dažniausiai turi kriptografinius duomenis, kurie suteikia galimybę autentifikuoti DID subjektą. DID pavyzdys pavaizduotas 10 pav.



10 pav. **DID pavyzdys [RSLA19]**

DID sudaro: schema, metodas, simbolių eilutė ir parametrai.

4.1.1. Metodas

Metodas turi atitikti šio reguliariojo reiškinio schemą: /^[a-z]+$/, kuri reiškia, kad metodą gali sudaryti tik mažosios raidės. Taip pat, metodas turi būti unikalus tarp visų decentralizuotų identifikatorių. Visi DID metodai yra patalpinti registre[2]. W3C rekomenduoja, kad metodą sudarytų mažiau nei 6 raidės ir jis būtų žmogui lengvai įsimenamas. Sukurtame SST sistemos prototipe naudotojas gali pasirinkti decentralizuoto identifikatoriaus metodą iš registro. DID kūrimo metu metodas yra tikrinamas pagal reguliariojo reiškinio schemą – funkcinis reikalavimas FR.9. W3C specifikaciją įgyvendinantis sprendimas Sovrin [SF17], leidžia naudotojui kurti DID tik naudojant *sov* metodą.

4.1.2. Simbolių eilutė

Simbolių eilutė turi atitikti šio reguliariojo reiškinio schemą: /^[0-9A-Za-z]*$/, kuri reiškia, kad metodą gali sudaryti tik mažosios ir didžiosios raidės, bei skaičiai. Simbolių eilutė turi būti unikali tarp visų metodo simbolių eilučių. Sukurtame SST sistemos prototipe kuriant DID simbolių

[2] https://w3c-ccg.github.io/did-method-registry/

eilutė yra generuojama automatiškai naudojant pirmos versijos UUID pagal kompiuterio MAC adresą ir laiko žymą – funkcinis reikalavimas FR.34. W3C specifikacijoje nėra nurodyta kokiu būdu simbolių eilutė turėtų būti kuriama, taip sukeliant kolizijos tikimybę ir DID unikalumo praradimą.

4.1.3. **Parametrai**

DID URL palaiko paprastą generalizuojamą parametrų formatą, paremtą matricine sintakse. Pagrindiniai DID parametrų vardai (pvz. paslaugų teikėjo pasirinkimui) yra nepriklausomi nuo DID metodo ir turi taip pat veikti visuose DID. Kiti parametrai (pvz. versijavimui) gali būti palaikomi tik tam tikrų DID metodų, bet turi veikti vienodai tuose DID, kuriuose jie yra palaikomi. Sukurtame SST sistemos prototipe DID užklausos parametrai yra kuriami taip pat naudojant paprastą generalizuojamą parametrų formatą – funkcinis reikalavimas FR.35. W3C specifikacijoje nėra nurodytą, kokiu formatu turėtų būti perduodi masyvo tipo parametrai, kas sukeltų neteisingą parametrų nuskaitymą skirtingų formatų sistemose.

4.1.4. **DID dokumentas**

Kiekvienas DID yra susietas su specialiai jam sugeneruotu DID dokumentu, kuriame yra saugomas duomenų modelis. DID dokumentas yra kuriamas JSON-LD formatu [SLKL18], kuris naudojamas saugoti dokumentus su susietais duomenimis. DID dokumentą gali sudaryti 8 pagrindinės dalys:

- Kontekstas (angl. *context*).
- DID subjektas (angl. *DID subject*).
- Vieši raktai (angl. *public keys*).
- Autentifikavimas (angl. *authentication*).
- Paslaugų nuorodos (angl. *service endpoints*).
- Sukūrimo data (angl. *creation date*).
- Atnaujinimo data (angl. *updation date*).
- Įrodymas (angl. *proof*).

4.1.5. **Kontekstas**

W3C specifikacija [RSLA19] naudoja @context lauką, kad nustatytų komunikacijos kontekstą. @context lauko reikšmė gali būti vienas arba daugiau URI, kur pirmo URI reikšmė turi būti: https://www.w3.org/ns/did/v1. Jeigu yra nurodytas daugiau negu vienas URI, konteksto lauko reikšmė turi būti interpretuojama kaip rikiuotas sąrašas. Rekomenduojama, kad URI turėtų nuorodą

į JSON-LD dokumentą, kuriame būtų programinei įrangai suprantama informacija apie kontekstą. DID metodo specifikacijos gali nurodyti savo pačių JSON-LD kontekstus, tačiau to daryti nerekomenduojama, nebent tai būtina norint tinkamai realizuoti DID metodą. Specifiniai DID metodo konteksto laukai turi neperdengti laukų nurodytų bendrame DID kontekse. @context lauko ištrauka yra pavaizduota 11 pav. DID dokumentas turi tenkinti šias sąlygas:

- DID dokumentas turi turėti vieną konteksto lauką šakninėje dokumento dalyje.
- Šio lauko raktas turi būti *@context.*
- Šio lauko reikšmė turi turėti decentralizuoto identifikatoriaus konteksto URI: https://www.w3.org/2019/did/v1

```
{
   "@context": "https://w3id.org/did/v1"
}
```

11 pav. **Konteksto lauko pavyzdys [RSLA19]**

Sukurtame SST sistemos prototipe kuriant DID dokumentą @context lauko reikšmė yra masyvas sudarytas iš https://www.w3.org/ns/did/v1 ir naudotojo pasirinktinai įvedamais URI, kurie nukreipia į JSON-LD dokumentą – funkcinis reikalavimas FR.36.

4.1.6. **DID subjektas**

DID subjektas yra DID dokumento identifikatorius, kuris nurodo, kuriam DID priklauso DID dokumentas. DID subjekto lauko ištrauka yra pavaizduota 12 pav. DID dokumentas turi tenkinti šias sąlygas:

- DID dokumentas turi turėti vieną DID subjektą.
- Šio lauko raktas turi būti *id*.
- Šio lauko reikšmė turi būti validus decentralizuotas identifikatorius.
- Kai DID dokumentas yra registruojamas decentralizuotų identifikatorių registre, užregistruotas DID turi sutapti su DID subjekto reikšme.

```
{
    "id": "did:example:123456789abcdefghi"
}
```

12 pav. **DID subjekto lauko pavyzdys [RSLA19]**

Sukurtame SST sistemos prototipe kuriant DID dokumentą naudotojas subjekto lauko reikšmę gali pasirinkti iš anksčiau sugeneruotų DID arba generuoti naują – funkcinis reikalavimas FR.37.

4.1.7. Vieši raktai

Vieši raktai yra naudojami skaitmeniniams parašams, kodavimui ir kitoms kriptografinėms operacijoms atlikti, kurių vienos iš pagrindinių funkcijų yra autentifikavimas ir saugios komunikacijos užtikrinimas su kitais paslaugų tiekėjais. Papildomai vieši raktai gali būti naudojami autorizavimo metu atliekant DID dokumento kūrimo, skaitymo, redagavimo ir trynimo operacijas. Tai gali būti nurodoma DID metodo specifikacijose.

Jeigu viešo rakto nėra DID dokumente, reiškia, kad raktas buvo panaikintas arba nėra validus. Specifikacija nurodo, kad DID dokumente gali būti įtraukti panaikinti raktai. DID dokumentas, kuris turi panaikintus viešus raktus taip pat turi turėti ir nuorodą į rakto panaikinimo informaciją. Kiekvieno DID metodo specifikacija turėtų nurodyti kaip raktų panaikinimas yra vykdomas ir saugomas. Viešų raktų lauko ištrauka yra pavaizduota 13 pav. DID dokumentas turi tenkinti šias sąlygas:

- DID dokumentas gali turėti *publicKey* lauką.
- *publicKey* lauko reikšmė turi būti viešų raktų masyvas.
- Kiekvienas viešas raktas turi turėti *id* ir *type* laukus ir vieną reikšmės lauką. Viešų raktų masyvas neturėtų turėti pasikartojančių raktų su ta pačia *id* reikšme ir skirtingais reikšmės laukais su skirtingais formatais.
- Kiekvienas viešas raktas turi turėti *controller* lauką, kuris nurodo privataus rakto savininką.
- Viešo rakto reikšmės lauko raktas turi būti vienas iš: *publicKeyPem, publicKeyJwk, publicKeyHex, publicKeyBase64, publicKeyBase58, publicKeyMultibase,* priklausomai nuo viešo rakto kodavimo ir formato.
- Raktų tipų ir formatų registras yra pasiekiamas adresu: https://w3c-ccg.github.io/ld-cryptosuite-registry/.

```
{
    "publicKey": [
        {
            "id": "did:example:123456789abcdefghi#keys-1",
            "type":  "RsaVerificationKey2018",
            "controller": "did:example:123456789abcdefghi",
            "publicKeyPem": "-----BEGIN PUBLIC KEY...END PUBLIC KEY-----\r\n"
        },
        {
            "id": "did:example:123456789abcdefghi#keys-2",
            "type": "Ed25519VerificationKey2018",
            "controller": "did:example:pqrstuvwxyz0987654321",
            "publicKeyBase58": "H3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
        },
        {
            "id": "did:example:123456789abcdefghi#keys-3",
            "type": "Secp256k1VerificationKey2018",
            "controller": "did:example:123456789abcdefghi",
            "publicKeyHex": "02b97c30de767f084ce3080168ee293053ba33b235d7116a3263d"
        }
    ]
}
```


13 pav. **Viešo rakto lauko pavyzdys [RSLA19]**

Algoritmas, naudojamas administruojant *publicKey* lauką DID dokumente:

- Tegu *value* yra duomenys susiję su *publicKey* lauku ir inicijavimo metu jų reikšmė yra lygi *null*.
- Jeigu *value* yra objektas, tai raktas yra įterptas. *PublicKey* lauko reikšmė yra lygi value reikšmei.
- Jeigu *value* yra simbolių eilutė, tai raktas yra įterptas kaip nuoroda. *Value* yra URL.
    - Iš URL gauname visus *publicKey* laukus susijusius su nurodyta *value* reikšme.
    - Iteruojame per visus viešuosius raktus, kol nurodyta *value* reikšmė yra lygi viešo rakto *id* laukui.
- Jeigu rezultatas neturi bent vieno iš *id*, *type* arba *controller* laukų, dokumentas yra nevalidus.

4.1.8. **Autentifikavimas**

Autentifikavimas yra mechanizmas, kuriuo galima kriptografiškai įrodyti, kad DIDsubjektas susijęs su tam tikru DID. Svarbu paminėti, kad autentifikavimas yra atskira dalis nuo autorizavimo, dėl to, kad DID subjektas gali suteikti leidimą kitiems redaguoti jo DID dokumentą (pvz. privataus rakto atstatymo atveju) neperleidžiant pilnos kontrolės. Autentifikavimo lauko ištrauka yra pavaizduota 14 pav. DID dokumentas turi tenkinti šias sąlygas:

- DID dokumentas gali turėti *authentication* lauką.
- *Authentication* lauko reikšmė turėtų būti verifikavimo metodų masyvas.
- Kiekvienas verifikavimo būdas gali būti įterptas kaip objektas arba nuoroda.

```
{
    "authentication": [
        "did:example:123456789abcdefghi#keys-1",
        {
            "id": "did:example:123456789abcdefghi#keys-2",
            "type": "Ed25519VerificationKey2018",
            "controller": "did:example:123456789abcdefghi",
            "publicKeyBase58": "H3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
        }
    ]
}
```

14 pav. **Autentifikavimo lauko pavyzdys [RSLA19]**

4.1.9. Paslaugų nuorodos

Be autentifikavimo ir autorizavimo kitas pagrindinis DID dokumento tikslas yra sudaryti galimybę atrasti DID subjektui reikalingas paslaugų nuorodas. Paslaugos nuoroda gali vaizduoti, bet kokio tipo paslaugą, su kuria DID subjektas nori komunikuoti, įskaitant decentralizuotos tapatybės valdymo sistemas, autentifikavimą, autorizavimą ar komunikaciją jose. Paslaugų nuorodų lauko ištrauka yra pavaizduota 15 pav. DID dokumentas turi tenkinti šias sąlygas:

- DID dokumentas gali turėti *service* lauką.
- *Service* lauko reikšmė turėtų būti paslaugų nuorodų masyvas.
- Kiekvienas paslaugų nuorodos objektas turi turėti *id*, *type* ir *serviceEndpoint* laukus bei taip pat gali turėti papildomus laukus.
- Paslaugų nuorodos protokolas turėtų būti patalpintas viešo standarto specifikacijoje.
- *ServiceEndpoint* lauko reikšmė turi būti JSON-LD objektas arba validus URI.

```
{
    "service": [
        {
            "id": "did:example:123456789abcdefghi#openid",
            "type": "OpenIdConnectVersion1.0Service",
            "serviceEndpoint": "https://openid.example.com/"
        },
        {
            "id": "did:example:123456789abcdefghi#vcr",
            "type": "CredentialRepositoryService",
            "serviceEndpoint": "https://repository.example.com/service/8377464"
        },
        {
            "id": "did:example:123456789abcdefghi#xdi",
            "type": "XdiService",
            "serviceEndpoint": "https://xdi.example.com/8377464"
        }
    ]
}
```

15 pav. **Paslaugos lauko pavyzdys [RSLA19]**

4.1.10. **Sukūrimo data**

Identifikatorių įrašų standartiniai meta duomenys turi originalaus sukūrimo laiko žymę. Sukūrimo datos lauko ištrauka yra pavaizduota 16 pav. DID dokumentas turi tenkinti šias sąlygas:

- DID dokumentas gali turėti lauką vaizduojantį sukūrimo laiko žymę. Rekomenduojama įtraukti šį lauką į DID dokumentą.
- Lauko raktas turi būti *created*.
- Lauko rakto reikšmė turi būti validi data.
- Ši datos laiko reikšmė turi būti normalizuota į UTC 00:00 formatą.
- Metodų specifikacijos, kurios pasitiki DLTs, turėtų reikalauti laiko reikšmės, kuri yra po „median time past“, kai DLT palaiko šį funkcionalumą.

```
{
    "created": "2002-10-10T17:00:00Z"
}
```

16 pav. **Sukūrimo datos lauko pavyzdys [RSLA]**

Sukurtame SST sistemos prototipe, kuriant DID dokumentą, sukūrimo data yra saugoma UTC formatu – nefunkcinis reikalavimas NFR.6.

4.1.11. **Atnaujinimo data**

Identifikatorių įrašų standartiniai meta duomenys turi paskutinio atnaujinimo laiko žymę. Redagavimo datos lauko ištrauka yra pavaizduota 17 pav. DID dokumentas turi tenkinti šias sąlygas:

- DID dokumentas gali turėti lauką vaizduojantį paskutinio atnaujinimo laiko žymę. Rekomenduojama įtraukti šį lauką į DID dokumentą.
- Lauko raktas turi būti *updated*.
- Šio rakto reikšmė turi būti validi pagal tas pačias taisykles, kaip ir *created* lauko reikšmė.

```
{
    "updated": "2016-10-17T02:41:00Z"
}
```

17 pav. **Atnaujinimo datos lauko pavyzdys [RSLA19]**

Sukurtame SST sistemos prototipe, kuriant DID dokumentą, atnaujinimo data yra saugoma UTC formatu – nefunkcinis reikalavimas NFR.7.

4.1.12. Įrodymas

DID dokumente *proof* laukas yra kriptografinis DID dokumento vientisumo įrodymas remiantis DID subjektu arba delegatu. Svarbu paminėti, kad šis įrodymas nėra kriptografinis sąryšio tarp DID ir DID dokumento įrodymas. Įrodymo laukas yra pavaizduotas 18 pav. DID dokumentas turi tenkinti šias sąlygas:

- DID dokumentas gali turėti vieną lauką vaizduojantį kriptografinį įrodymą.
- Šio lauko raktas turi būti *proof*.
- Rakto reikšmė turi būti validus JSON-LD dokumento įrodymas.

```
{
    "proof": {
        "type": "LinkedDataSignature2015",
        "created": "2016-02-08T16:02:20Z",
        "creator": "did:example:8uQhQMGzWxR8vw5P3UWH1ja#keys-1",
        "signatureValue": "QNB13Y7Q9...1tzjn4w=="
    }
}
```

18 pav. **Įrodymo lauko pavyzdys [RSLA19]**

4.2. DID validatorius

DID validatorius (angl. DID resolver) yra programinės įrangos komponentas, kuris atlieka DID validavimą pagal įvestą DID. Priklausomai nuo metodo jo realizacija gali skirtis. Pagal W3C specifikaciją DID validatorius privalo palaikyti bent 1 DID validavimo algoritmą, Pagrindinė DID validatoriaus funkcija yra pagal nurodytą DID grąžinti DID dokumentą. Sukurtame SST sistemos prototipe DID validatorius yra integruotas į SDK biblioteką.

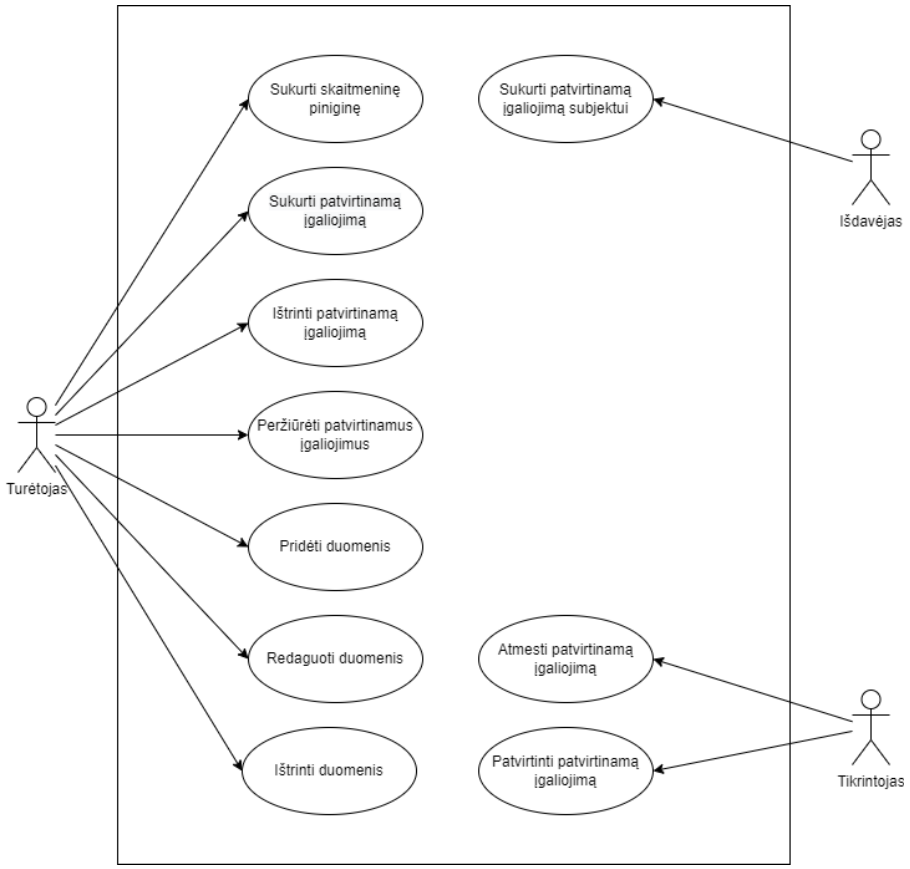
5. Savarankiškos suverenios tapatybės realizacija

Šiame skyriuje bus pateikta sistemos realizacija remiantis 1.1. skyriuje nurodytu savarankiškos suverenios tapatybės komponentų modeliu. Kiekvienas iš komponentų yra išsamiai aprašytas 2–4 skyriuose. Remiantis W3C specifikacijomis [RSLA19, SLC19] identifikuoti sistemos naudojimo atvejai ir sukurti funkciniai ir nefunkciniai reikalavimai. Remiantis sistemos naudojimo atvejais ir funkciniais bei nefunkciniais reikalavimais sukurta sistemos architektūra.

5.1. Naudojimo atvejai

Sistema turi 3 roles: turėtojas, išdavėjas, tikrintojas. Kiekvienai iš rolių yra apibrėžti

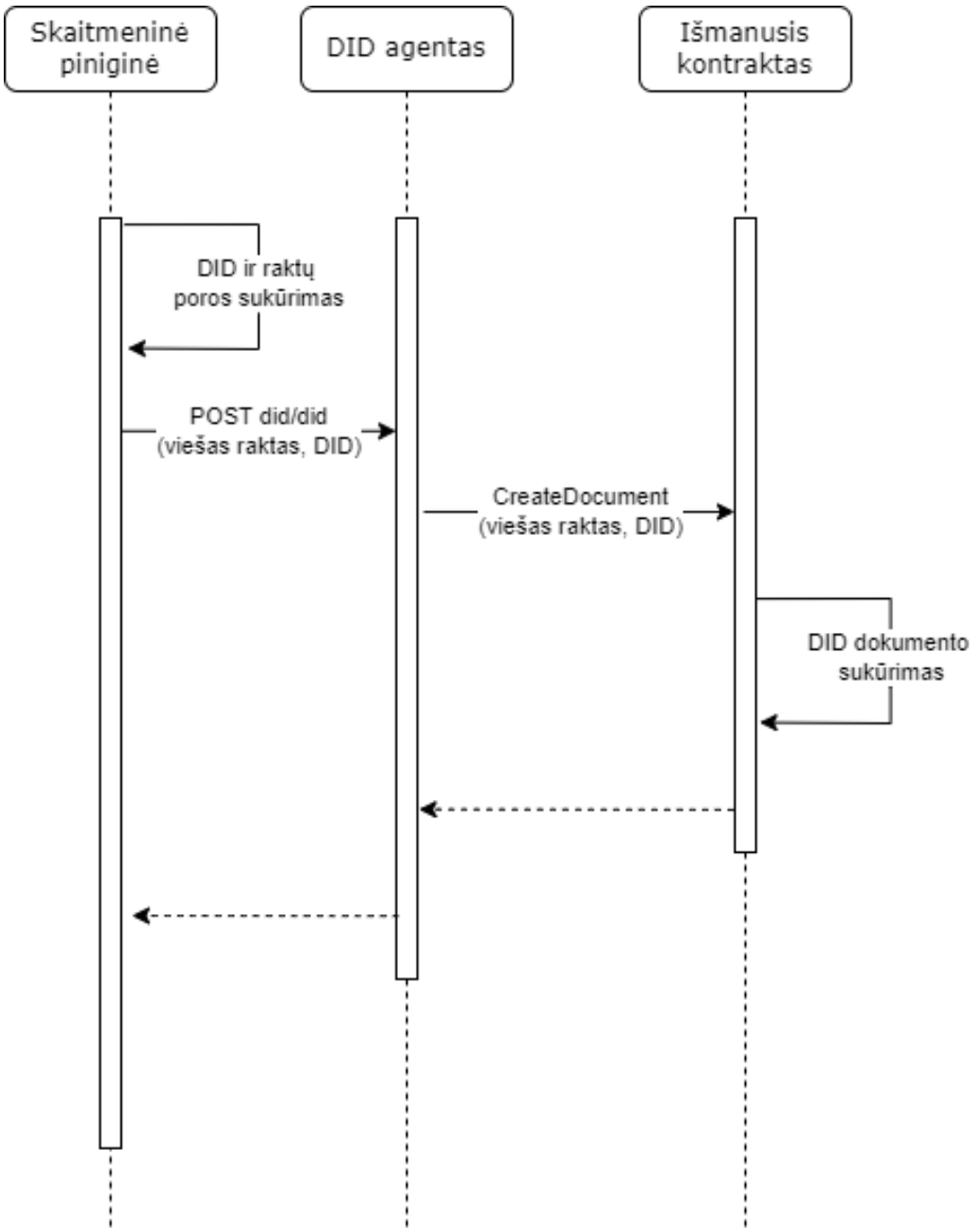
naudojimo atvejai remiantis W3C patvirtinamų įgaliojimų naudojimo atvejų specifikacija [OLSBSE19]. Svarbu paminėti, kad visi sistemos naudotojai gali atlikti visų rolių funkcijas. SST sistemos naudojimo atvejų UML diagrama pavaizduota 19 pav. Lentelėse nr. 3–12 yra pateikti naudojimo atvejų aprašymai. Naudojimo atvejų sekos diagramos pavaizduotos 20–27 pav.



19 pav. **SST sistemos naudojimo atvejai**

3 lentelė. **Skaitmeninės piniginės sukūrimas**

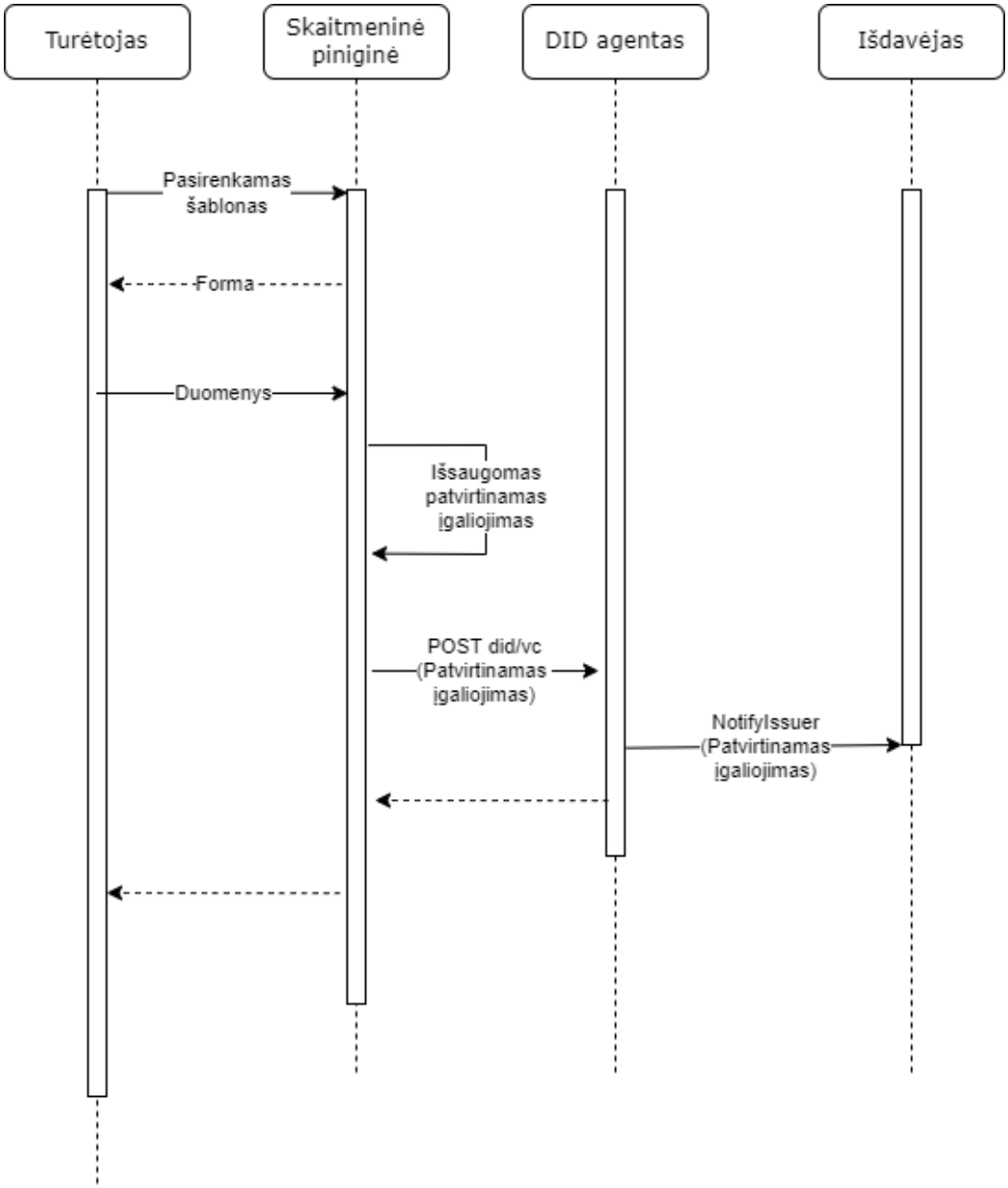
Pavadinimas	Skaitmeninės piniginės sukūrimas
Aprašymas	Sukurti skaitmeninę piniginę
Aktorius	Turėtojas, išdavėjas, tikrintojas
Sąlygos prieš	–
Sąlygos po	• Sukuriama viešo ir privataus raktų pora • Sukuriamas DID • Sukuriamas DID dokumentas • DID dokumentas patalpinamas išskirstytos apskaitos technologijoje
Trigeriai	• Poreikis susikurti skaitmeninę piniginę
Veiksmai	• Aktorius paspaudžia skaitmeninės piniginės sukūrimo mygtuką



20 pav. **Skaitmeninės piniginės sukūrimo sekos diagrama**

4 lentelė. **Patvirtinamo įgaliojimo sukūrimas**

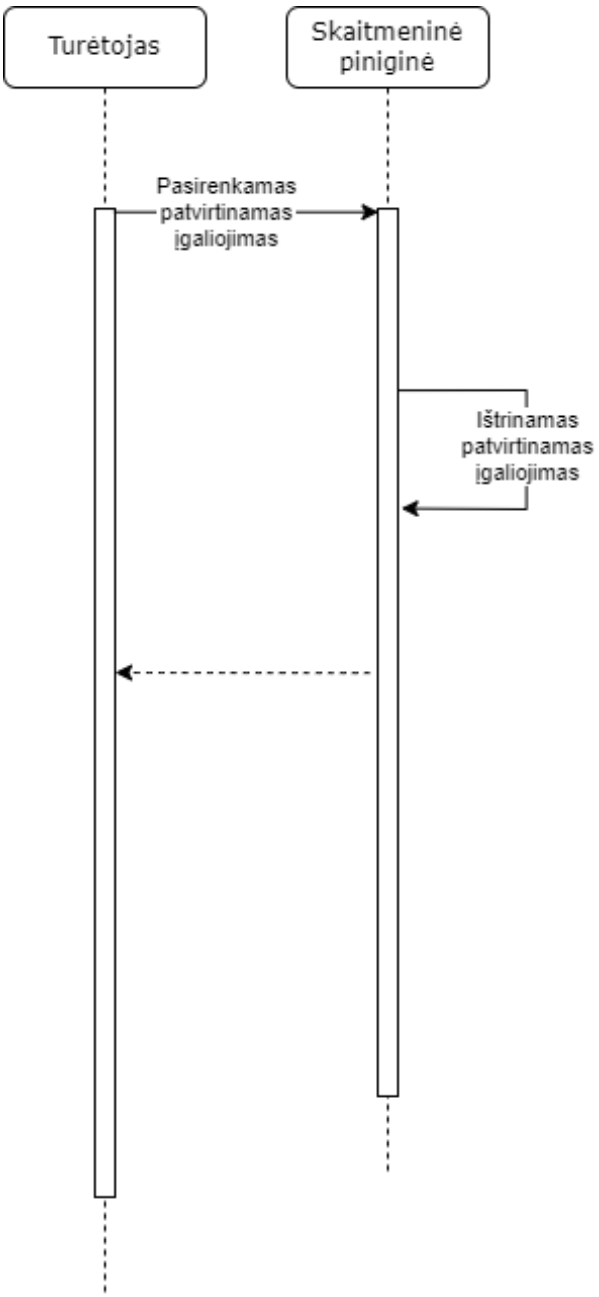
Pavadinimas	Patvirtinamo įgaliojimo sukūrimas
Aprašymas	Sukurti patvirtinamą įgaliojimą
Aktorius	Turėtojas
Sąlygos prieš	• Aktorius yra susikūręs skaitmeninę piniginę
Sąlygos po	• Sukuriamas patvirtinamas įgaliojimas • Išdavėjas informuojamas apie sukurtą patvirtinamą įgaliojimą
Trigeriai	• Aktorius paspaudžia patvirtinamo įgaliojimo sukūrimo mygtuką
Veiksmai	• Aktorius paspaudžia patvirtinamo įgaliojimo sukūrimo mygtuką • Aktorius pasirenka patvirtinamo įgaliojimo šabloną • Aktorius įveda tvirtinamus duomenis • Aktorius pasirenka išdavėją



21 pav. **Patvirtinamo įgaliojimo sukūrimo sekos diagrama**

5 lentelė. **Patvirtinamo įgaliojimo ištrynimas**

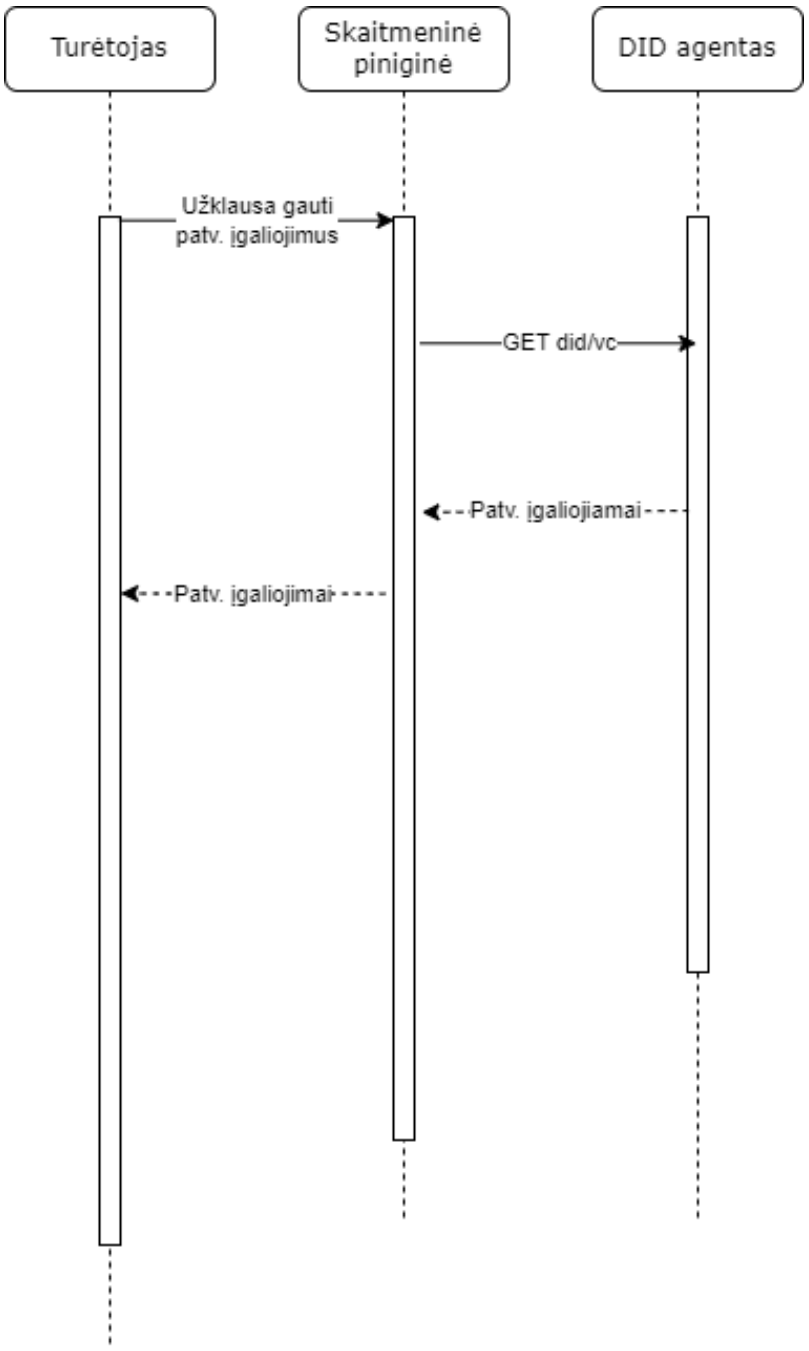
Pavadinimas	Patvirtinamo įgaliojimo ištrynimas
Aprašymas	Ištrinti pasirinktą patvirtinamą įgaliojimą
Aktorius	Turėtojas
Sąlygos prieš	• Aktorius yra susikūręs skaitmeninę piniginę • Aktorius yra pasirinkęs patvirtinamą įgaliojimą
Sąlygos po	• Ištrinamas patvirtinamas įgaliojimas
Trigeriai	• Aktorius paspaudžia patvirtinamo įgaliojimo ištrynimo mygtuką
Veiksmai	• Aktorius pasirenka patvirtinamą įgaliojimą • Aktorius paspaudžia ištrynimo mygtuką • Aktorius patvirtina ištrynimą iššokančiame lange



22 pav. **Patvirtinamo įgaliojimo ištrynimo sekos diagrama**

6 lentelė. **Patvirtinamų įgaliojimų peržiūra**

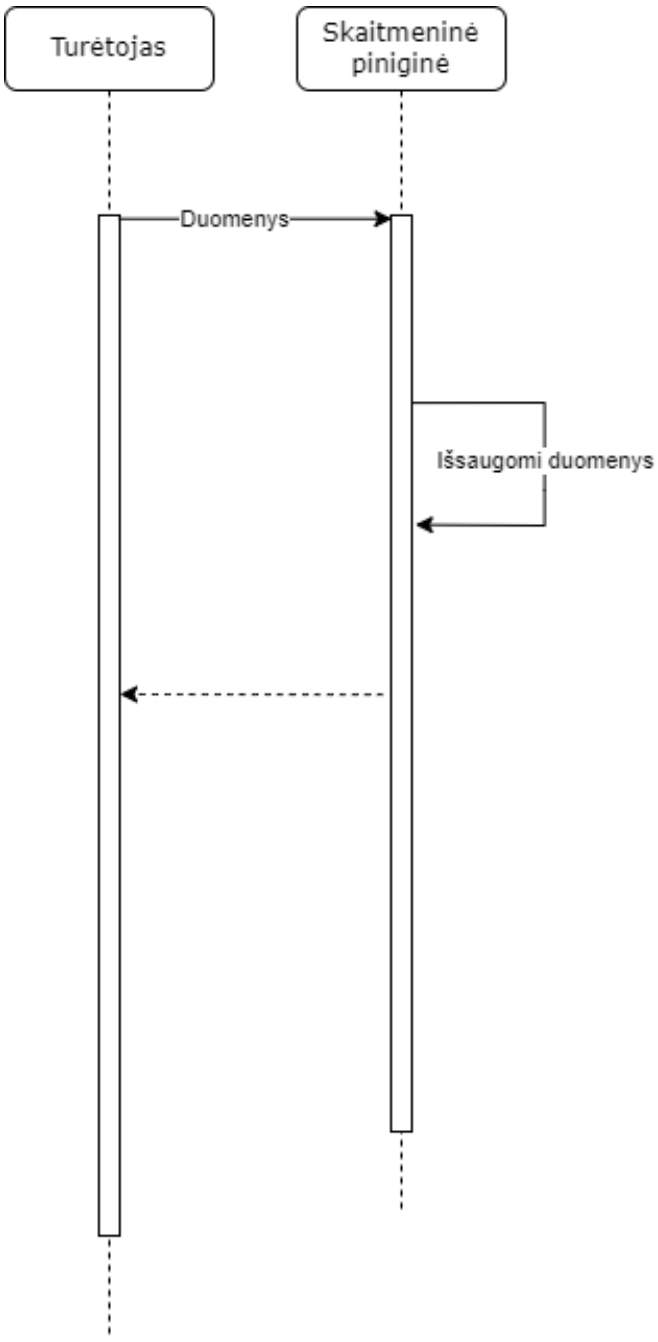
Pavadinimas	Patvirtinamų įgaliojimų peržiūra
Aprašymas	Peržiūrėti patvirtinamus įgaliojimus
Aktorius	Turėtojas
Sąlygos prieš	• Aktorius yra susikūręs skaitmeninę piniginę • Aktorius turi sukurtų patvirtinamų įgaliojimų
Sąlygos po	• Atvaizduojamas patvirtinamų įgaliojimų sąrašas
Trigeriai	• Aktorius atidaro patvirtinamų įgaliojimų langą
Veiksmai	• Aktorius atsidaro patvirtinamų įgaliojimų langą



23 pav. **Patvirtinamų įgaliojimų peržiūros sekos diagrama**

7 lentelė. **Duomenų pridėjimas**

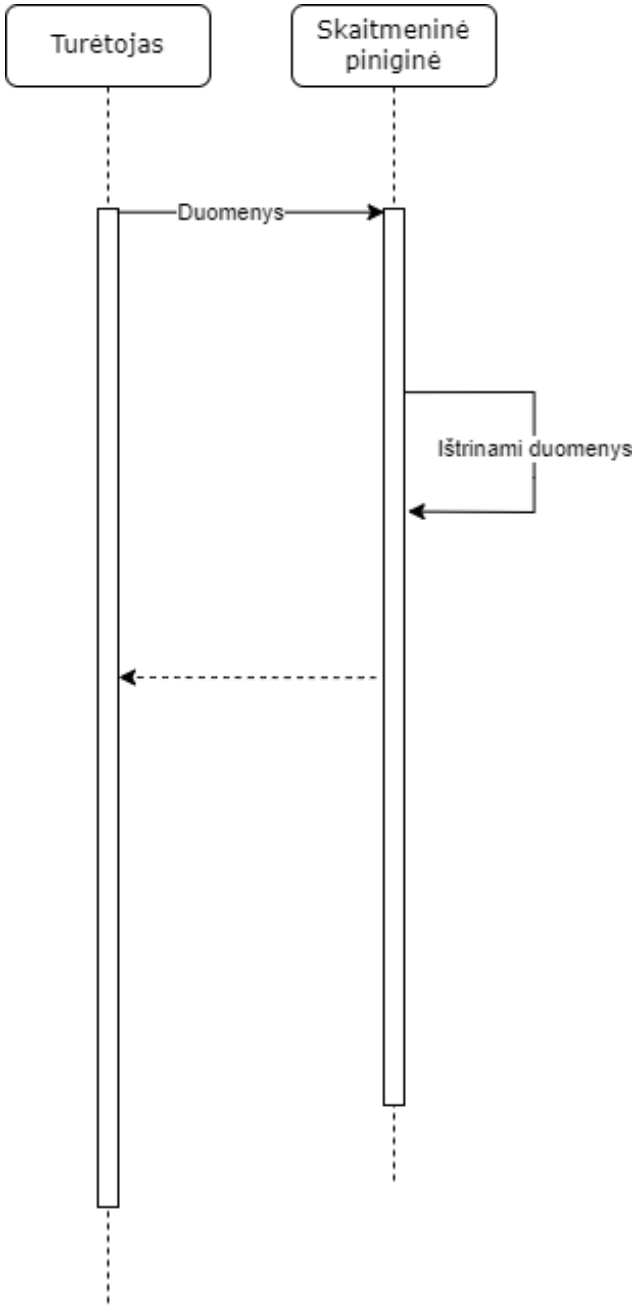
Pavadinimas	Duomenų pridėjimas
Aprašymas	Pridėti duomenis į skaitmeninę piniginę
Aktorius	Turėtojas
Sąlygos prieš	• Aktorius yra susikūręs skaitmeninę piniginę
Sąlygos po	• Duomenys pridedami prie skaitmeninės piniginės
Trigeriai	• Aktorius paspaudžia duomenų pridėjimo mygtuką
Veiksmai	• Aktorius atsidaro duomenų pridėjimo langą • Aktorius įveda pasirinktus duomenis • Aktorius paspaudžia duomenų pridėjimo mygtuką



24 pav. **Duomenų pridėjimo sekos diagrama**

8 lentelė. **Duomenų redagavimas**

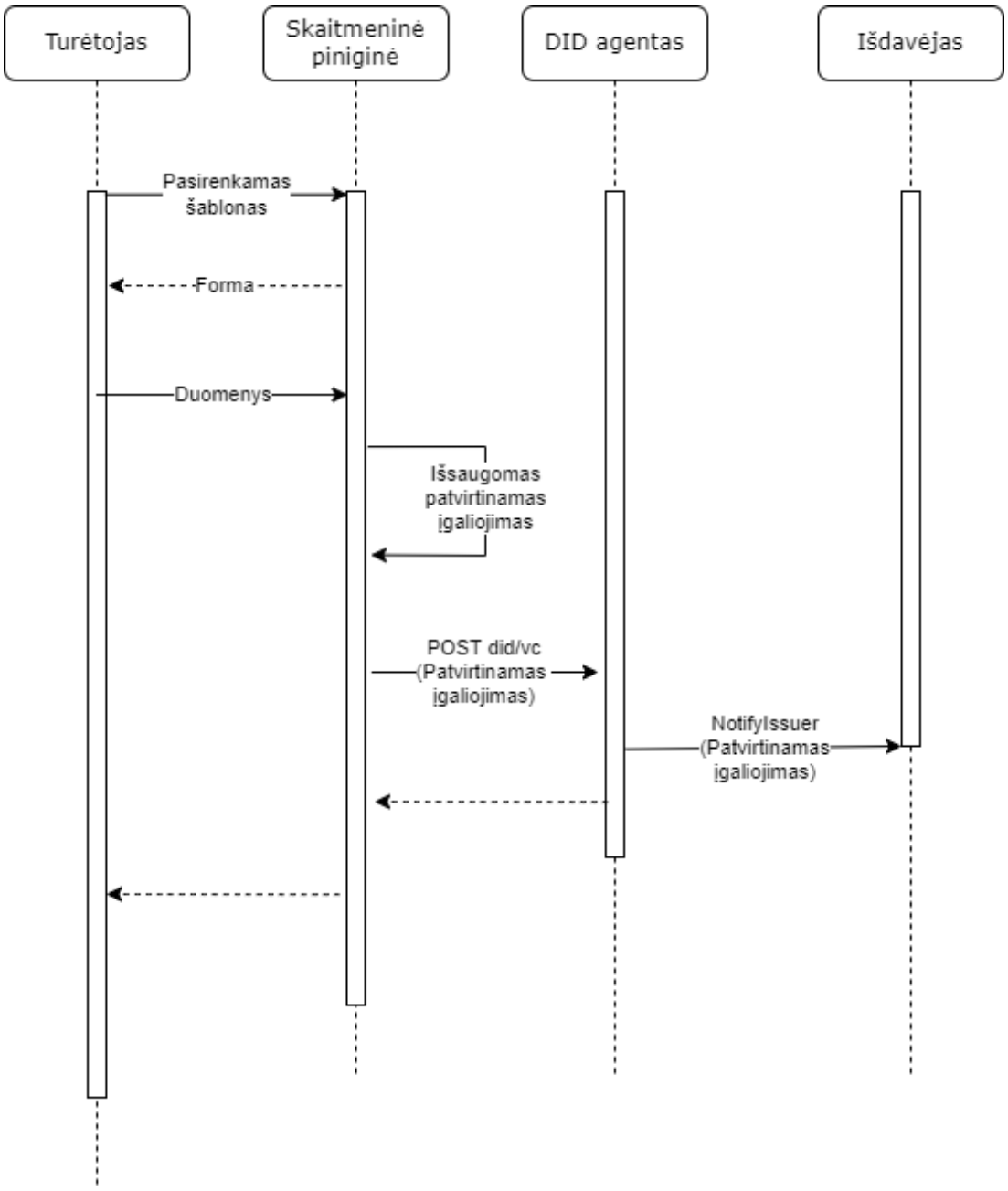
Pavadinimas	Duomenų redagavimas
Aprašymas	Redaguoti duomenis skaitmeninėje piniginėje
Aktorius	Turėtojas
Sąlygos prieš	• Aktorius yra susikūręs skaitmeninę piniginę
Sąlygos po	• Duomenys atnaujinami
Trigeriai	• Aktorius paspaudžia duomenų atnaujinimo mygtuką
Veiksmai	• Aktorius pasirenka redaguojamus duomenis • Aktorius atnaujina pasirinktus duomenis • Aktorius paspaudžia duomenų atnaujinimo mygtuką



25 pav. **Duomenų ištrynimo sekos diagrama**

9 lentelė. **Duomenų ištrynimas**

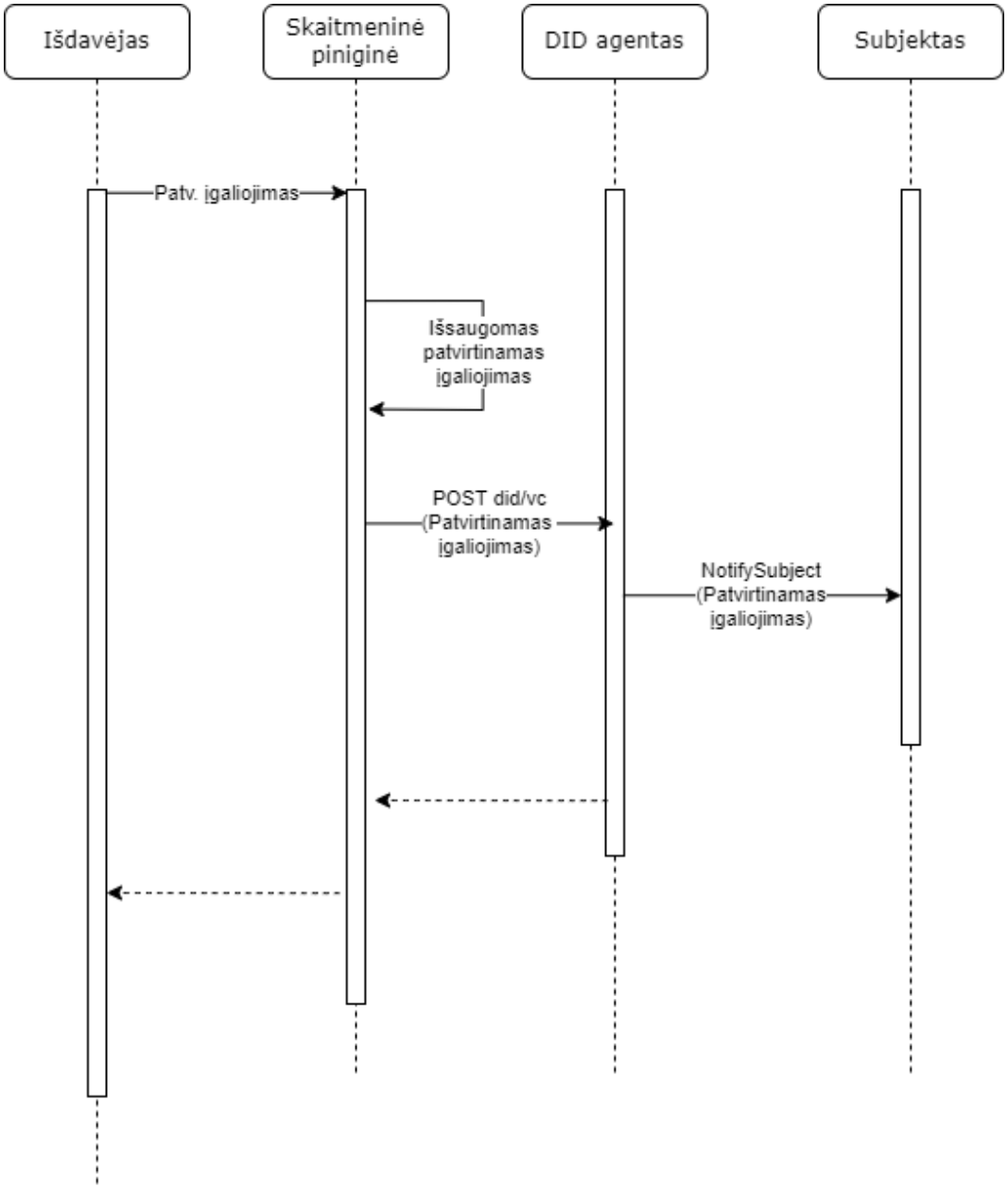
Pavadinimas	Duomenų ištrynimas
Aprašymas	Ištrinti duomenis skaitmeninėje piniginėje
Aktorius	Turėtojas
Sąlygos prieš	• Aktorius yra susikūręs skaitmeninę piniginę
Sąlygos po	• Duomenys ištrinami
Trigeriai	• Aktorius paspaudžia duomenų ištrynimo mygtuką
Veiksmai	• Aktorius pasirenka duomenis, kuriuos nori ištrinti • Aktorius paspaudžia duomenų ištrynimo mygtuką • Aktorius paspaudžia ištrynimo mygtuką iššokančiame lange



26 pav. **Patvirtinamo įgaliojimo sukūrimo subjektui sekos diagrama**

10 lentelė. **Patvirtinamo įgaliojimo sukūrimas subjektui**

Pavadinimas	Patvirtinamo įgaliojimo sukūrimas subjektui
Aprašymas	Sukurti patvirtinamą įgaliojimą subjektui
Aktorius	Išdavėjas
Sąlygos prieš	• Aktorius yra susikūręs skaitmeninę piniginę
Sąlygos po	• Sukuriamas patvirtinamas įgaliojimas • Subjektas informuojamas apie sukurtą patvirtinamą įgaliojimą
Trigeriai	• Aktorius paspaudžia patvirtinamo įgaliojimo sukūrimo mygtuką
Veiksmai	• Aktorius paspaudžia patvirtinamo įgaliojimo sukūrimo mygtuką • Aktorius pasirenka patvirtinamo įgaliojimo šabloną • Aktorius įvedami duomenis • Aktorius paspaudžia išsaugojimo mygtuką



27 pav. **Patvirtinamo įgaliojimo sukūrimas subjektui sekos diagrama**

11 lentelė. **Patvirtinamo įgaliojimo patvirtinimas**

Pavadinimas	Patvirtinamo įgaliojimo patvirtinimas
Aprašymas	Patvirtinti patvirtinamą įgaliojimą
Aktorius	Tikrintojas
Sąlygos prieš	• Aktorius yra susikūręs skaitmeninę piniginę • Aktorius pasirenka tvirtinamą patvirtinamą įgaliojimą
Sąlygos po	• Patvirtinamas įgaliojimas yra pasirašomas aktoriaus privačiu raktu • Turėtojas informuojamas apie patvirtintą patvirtinamą įgaliojimą
Trigeriai	• Aktorius paspaudžia patvirtinamo įgaliojimo patvirtinimo mygtuką
Veiksmai	• Aktorius pasirenka tvirtinamą patvirtinamą įgaliojimą • Aktorius paspaudžia patvirtinimo mygtuką • Subjektas yra informuojamas apie patvirtiną patvirtinamą įgaliojimą

12 lentelė. **Patvirtinamo įgaliojimo atmetimas**

Pavadinimas	Patvirtinamo įgaliojimo atmetimas
Aprašymas	Atmesti patvirtinamą įgaliojimą
Aktorius	Tikrintojas
Sąlygos prieš	• Aktorius yra susikūręs skaitmeninę piniginę • Aktorius pasirenka tvirtinamą VC
Sąlygos po	• Turėtojas informuojamas apie atmestą patvirtinamą įgaliojimą
Trigeriai	• Aktorius paspaudžia patvirtinamo įgaliojimo atmetimo mygtuką
Veiksmai	• Aktorius pasirenka tvirtinamą VC • Aktorius paspaudžia atmetimo mygtuką • Subjektas yra informuojamas apie atmestą patvirtinamą įgaliojimą

5.2. Naudojimo atvejo pavyzdys

Naudojimo atvejo pavyzdyje nagrinėjama, kaip SST kartu su patvirtinimais įgaliojimais (VC), decentralizuotais identifikatoriais (DID) ir išskirstytos apskaitos technologija (DLT) gali užtikrinti vairuotojo pažymėjimo patikrinimą. Scenarijus: vairuotojas (**turėtojas**), nori įrodyti savo teisę vairuoti automobilį automobilių nuomos įmonei (**tikrintojas**) neatskleisdamas perteklinės asmeninės informacijos. Automobilių nuomos įmonė į savo patikros procesą integravo SST technologiją, leidžiančią vairuotojui pateikti skaitmeninį vairuotojo pažymėjimą autentiškumui patvirtinti. Sudedamosios dalys:

- **Patvirtinami įgaliojimai**: Vairuotojo pažymėjimas yra išduodamas kaip patvirtinamas įgaliojimas, kuriame yra reikalinga informacija: vardas, pavardė, vairuotojo pažymėjimo numeris ir galiojimo data. Vairuotojo pažymėjimą kriptografiškai pasirašo vairuotojus registruojanti įmonė (**išdavėjas**), užtikrindama vientisumą ir autentiškumą.
- **Decentralizuoti identifikatoriai**: Vairuotojas turi DID, susietą su SST sistema, kuris naudojamas kaip unikalus vairuotojo identifikatorius. DID suteikia vairuotojui kontrolę, nes tik jis gali valdyti ir kontroliuoti savo DID, nepasikliaujant centralizuota institucija.
- **Išskirstytos apskaitos technologija**: DLT, naudojama saugoti ir valdyti decentralizuotą SST palaikančią infrastruktūrą.

Naudojimo atvejo procesas:

- Vairuotojas inicijuoja vairuotojo pažymėjimo patikrinimo procesą.
- Automobilių nuomos įmonė, kaip tikrintojas, prašo vairuotojo pažymėjimo informacijos.
- Vairuotojas naudodamas savo SST piniginę, kurioje saugomi patvirtinami įgaliojimai, pasirenka ir pateikia savo skaitmeninį vairuotojo pažymėjimą.

- Automobilių nuomos įmonė, veikdama kaip tikrintojas, patikrina vairuotojo pažymėjimo autentiškumą ir vientisumą. Ji gauna vairuotojo pažymėjimo DID.
- Automobilių nuomos įmonė išsiunčią užklausą į DLT, kad gautų vairuotojo ir vairuotojus registruojančios įmonės viešuosius raktus ir patikrintų vairuotojo pažymėjimo skaitmeninius parašus.
- Automobilių nuomos įmonė gali patvirtinti Alisos vairuotojo pažymėjimo autentiškumą ir galiojimą be centralizuotos institucijos ar prieigos prie vairuotojo asmens duomenų.

Naudojant VC, DID ir DLT, vairuotojo pažymėjimo patikrinimas tampa saugiu ir privatumą išsaugančiu procesu. Šis naudojimo atvejis rodo, kaip SST gali palengvinti tapatybės valdymą ir autentiškumo nustatymą įvairiose srityse, užtikrinant naudotojų kontrolę ir duomenų vientisumą.

5.3. Funkciniai reikalavimai

Remiantis W3C specifikacijomis [RSLA19, SLC19] ir naudojimo atvejais sukurti SST sistemos funkciniai reikalavimai pavaizduoti 13 lentelėje.

13 lentelė. **Funkciniai reikalavimai**

Id	Reikalavimas
FR.1	Naudotojas turi galėti susikurti skaitmeninę piniginę
FR.2	Naudotojas turi galėti išsisaugoti privatų raktą išoriniame prietaise
FR.3	Sistema skaitmeninės piniginės kūrimo metu turi sugeneruoti viešo ir privataus raktų porą
FR.4	Sistema turi sugeneruoti naudotojui DID
FR.5	Sistema turi sugeneruoti naudotojui DID dokumentą
FR.6	DID dokumentas turi turėti DID ir sugeneruotą viešą raktą
FR.7	DID turi būti sudarytas iš: schemos, metodo, simbolių eilutės
FR.8	DID metodas turi būti unikalus tarp visų decentralizuotų identifikatorių
FR.9	DID metodas turi tenkinti reguliariojo reiškinio schemą: /^[a-z]+$/
FR.10	DID simbolių eilutė turi tenkinti reguliariojo reiškinio schemą: /^[0-9A-Za-z]*$/
FR.11	Naudotojas turi galėti pridėti duomenis prie skaitmeninės piniginės
FR.12	Naudotojas turi galėti redaguoti duomenis esančius skaitmeninės piniginės prietaiso atmintyje
FR.13	Naudotojas turi galėti ištrinti duomenis iš skaitmeninės piniginės
FR.14	Skaitmeninė piniginė duomenis turi saugoti prietaiso atmintyje
FR.15	Duomenys saugomi prietaiso atmintyje turi būti kriptografiškai apsaugoti
FR.16	Naudotojas turi galėti pasidalinti savo DID QR formatu
FR.17	Naudotojas turi galėti sukurti patvirtinamą įgaliojimą iš prietaiso atmintyje saugomų duomenų
FR.18	Naudotojas turi galėti pridėti papildomus duomenis kurdamas patvirtinamą įgaliojimą
FR.19	Sistema turi pridėti patvirtinamą įgaliojimą prie naudotojo pasirinkto DID dokumento
FR.20	Naudotojas turi galėti patvirtinti kito DID subjekto patvirtinamą įgaliojimą
FR.21	Naudotojas turi galėti atmesti kito DID subjekto patvirtinamą įgaliojimą
FR.22	DID dokumentas turi būti sudarytas iš: konteksto, subjekto, viešų raktų,

Id	Reikalavimas
	autentifikavimo, paslaugų nuorodų, sukūrimo datos, atnaujinimo datos, įrodymo
FR.23	DID dokumentas turi būti JSON-LD formato
FR.24	DID dokumento kontekstas turi būti rikiuotas sąrašas iš vieno arba daugiau URI nuorodų
FR.25	DID dokumento konteksto pirma reikšmė turi būti: https://www.w3.org/2019/did/v
FR.26	DID dokumento subjekto reikšmė turi būti validus DID
FR.27	DID dokumento vieši raktai turi būti viešų raktų masyvas
FR.28	DID dokumento viešas raktas turi turėti nuorodą į privataus rakto savininką
FR.29	DID dokumento viešo rakto galimi formatai*: publicKeyPem, publicKeyJwk, publicKeyHex, publicKeyBase64, publicKeyBase58, publicKeyMultibase*
FR.30	DID dokumento autentifikavimas turi būti verifikavimo metodų masyvas
FR.31	DID dokumento autentifikavimo reikšmė gali būti objektas arba nuorodą į objektą
FR.32	DID dokumento paslaugų nuorodos turi būti paslaugų nuorodų masyvas
FR.33	DID dokumento paslaugų nuorodą turi būti sudaryta iš: DID, tipo ir nuorodos į paslaugą
FR.34	DID simbolių eilutė yra generuojama automatiškai naudojant pirmos versijos UUID pagal kompiuterio MAC adresą ir laiko žymą
FR.35	DID užklausos parametrai yra kuriami taip pat naudojant paprastą generalizuojamą parametrų formatą
FR.36	@context lauko reikšmė yra masyvas sudarytas iš https://www.w3.org/ns/did/v1 ir naudotojo pasirinktinai įvedamais URI kurie nukreipia į JSON-LD dokumentą
FR.37	Kuriant DID dokumentą naudotojas subjekto lauko reikšmę gali pasirinkti iš anksčiau sugeneruotų DID arba generuoti naują

5.4. Nefunkciniai reikalavimai

Remiantis W3C specifikacijomis [RSLA19, SLC19] sukurti SST sistemos nefunkciniai reikalavimai pavaizduoti 14 lentelėje.

14 lentelė. **Nefunkciniai reikalavimai**

Id	Reikalavimas
NFR.1	Skaitmeninė piniginė turi būti palaikoma IOS ir Android operacinėse sistemose
NFR.2	Užklausos tarp skaitmeninės piniginės ir DID agento turi naudoti HTTPS protokolą
NFR.3	Įrenginio laisva atmintis turi būti ne mažesnė negu 16 MB ir nepasiekiama kitoms programoms
NFR.5	Sugeneruotos raktų poros turi būti RSA formato
NFR.6	DID dokumento sukūrimo data turi būti UTC 00:00 formato
NFR.7	DID dokumento atnaujinimo data turi būti UTC 00:00 formato
NFR.8	Duomenys įrenginyje turi būti laikomi saugiai
NFR.9	Atsakymas iš DID agento turi grįžti greičiau nei per 2 sekundes

5.5. Savarankiškos suverenios tapatybės architektūra

Savarankiškos suverenios tapatybės (SST) architektūra sukurta remiantis trimis pagrindinėmis technologijomis: išskirstytos apskaitos technologija (DLT), patvirtinamais įgaliojimais (VC) ir

decentralizuotais identifikatoriais (DID). Šiame skyriuje pateikiamas išsamus kiekvienos technologijos įgyvendinimo SST architektūroje aprašymas.

5.5.1. Išskirstytos apskaitos technologija

Išskirstytos apskaitos technologija yra pamatinis sluoksnis, užtikrinantis decentralizuotą ir skaidrų duomenų saugojimą, kuris yra SST architektūros pagrindas. Ji blokų grandinėje fiksuoja patvirtinamų įgaliojimų išdavimą, perdavimą ir patikrinimą. VC integravimas į DLT užtikrina įgaliojimų nekeičiamumą ir skaidrumą, didina pasitikėjimą ir sudaro sąlygas tikrinimo procesams. Naudodama kriptografines simbolių eilutes ir skaitmeninius parašus, DLT garantuoja patvirtinamų įgaliojimų vientisumą ir autentiškumą. Toliau pateikiama išsami informacija apie DLT įgyvendinimą SST architektūroje:

- **Konsensuso sluoksnis**: Konsensuso sluoksnis užtikrina išskirstytos apskaitos technologijos būsenos susitarimą. Įprasti sutarimo mechanizmai yra darbo įrodymas (PoW) ir statymo įrodymas (PoS).
- **Blokų grandinės sluoksnis**: Šį sluoksnį sudaro blokų grandinės tinklas, kuriame vyksta su tapatybės valdymu susiję sandoriai ir sąveikos. Jame saugomi patvirtinami įgaliojimai, sandorių istorija ir kiti transakcijų duomenys.
- **Išmaniųjų kontraktų sluoksnis**: Išmanieji kontraktai leidžia vykdyti su savarankiškais tapatybės duomenimis susijusius sandorius. Jie apibrėžia taisykles ir logiką, kaip sukurti, išduoti ir patikrinti patvirtinamus įgaliojimus.

5.5.2. Išmanusis kontraktas

SST architektūroje naudojamas „Solidity“ išmanusis kontraktas, kad į decentralizuotą tapatybės valdymo sistemą būtų integruoti VC, DLT ir DID. „Ethereum“ išmaniosioms sutartims sukurta „Solidity“ yra išbandyta ir paplitusi decentralizuotų programų kūrimo platforma. Jos funkcijos leidžia įgyvendinti sudėtingą logiką, kartu užtikrinant suderinamumą su Ethereum virtualia mašina (EVM).

SHA-256 kaip šifravimo algoritmas pasirinktas atsižvelgiant į jo kriptografinį stiprumą. SHA-256 yra plačiai naudojamas tikrinti duomenų vientisumui ir kurti skaitmeninius parašus. Šis algoritmas sukurtas taip, kad sugeneruotų unikalias fiksuoto dydžio simbolių eilutes, tinkamas VC.

Šių technologijų pasirinkimas atitinka SST architektūros saugumo reikalavimus. Naudojant „Solidity“, įgyvendinimui yra naudingas „Ethereum“ patikimumas ir brandumas, užtikrinantis patikimą ir palaikomą kūrimo aplinką. Naudojant SHA-256 kaip šifravimo algoritmą, užtikrinamas

VC vientisumas ir autentiškumas, apsaugant nuo duomenų klastojimo ir neleistinų pakeitimų.

Šių technologijų pasirinkimą lėmė naudojimas ir pripažinimas akademiniuose tyrimuose. Solidity buvo tiriamas ir naudojamas blokų grandinėmis grindžiamose taikomosiose programose, o SHA-256 yra patikimas šifravimo algoritmas, naudojamas kriptografiniuose protokoluose. Šių technologijų akademinis ir pramonės pripažinimas didina SST įgyvendinimo patikimumą ir saugumą, suderintą su geriausia praktika ir nustatytais standartais decentralizuoto tapatybės valdymo srityje.

5.5.3. **Patvirtinami įgaliojimai**

Patvirtinami įgaliojimai – tai skaitmeniniai tapatybės atributų ar duomenų paketai, kurie leidžia saugiai ir patikimai saugoti ir tvarkyti asmeninę informaciją. Toliau pateikiama išsami informacija apie patvirtinamų įgaliojimų realizaciją SST architektūroje:

- **Įgaliojimų išdavimo sluoksnis**: Šiame sluoksnyje patikimi subjektai kuria ir išduoda patvirtinamus įgaliojimus. Jie tikrina išdavėjo tapatybę ir autentiškumą bei užtikrina įgaliojimo vientisumą.
- **Įgaliojimų tikrinimo sluoksnis**: Šis sluoksnis atsakingas už patvirtinamų įgaliojimų autentiškumo ir vientisumo patikrinimą. Jis apima patvirtinamo įgaliojimo skaitmeninio parašo ir išdavėjo viešojo rakto kriptografinį patikrinimą.
- **Įgaliojimų saugojimo sluoksnis**: Patvirtinamieji įgaliojimai saugomi skaitmeninėse piniginėse arba saugyklose, taip užtikrinant privatumą ir prieigos kontrolę. Šiame sluoksnyje numatyti įgaliojimų saugojimo ir gavimo mechanizmai.

5.5.4. **Decentralizuoti identifikatoriai**

DID naudojami kaip visuotinai unikalūs asmenų, organizacijų ar daiktų identifikatoriai SST sistemose. Toliau pateikiama išsami informacija apie DID įgyvendinimą SST architektūroje:

- **DID kūrimo sluoksnis**: Šis sluoksnis sukuria ir priskiria unikalius DID asmenims ar subjektams. Jame naudojami kriptografiniai metodai, kad būtų užtikrintas DID unikalumas ir saugumas.
- **DID skirstymo sluoksnis**: Skirstymo sluoksnis leidžia gauti ir patikrinti DID. Jis priskiria DID atitinkamai tapatybės informacijai, pavyzdžiui, viešiesiems raktams arba paslaugų nuorodoms, ir taip užtikriną sklandžią sąveiką tarp skirtingų SST sistemų.
- **DID valdymo sluoksnis**: Šis sluoksnis palengvina DID valdymą, įskaitant atnaujinimo, atšaukimo ir atkūrimo procesus. Juo užtikrinamas DID vientisumas ir kontrolė, kurią atlieka su jais susijęs asmuo arba subjektas.

6. SST architektūros ir SST principų [CA16] palyginimas su sistemos reikalavimais

Lentelėje nr. 15 yra pateikiamas išsamus savarankiškos suverenios tapatybės architektūros ir principų, išdėstytų knygoje „Kelias į savarankišką tapatybę“ [CA16], palyginimas. Joje pabrėžiamas sistemos architektūros atitikimas kiekvienam principui ir pateikiami susiję sistemos funkciniai ir nefunkciniai reikalavimai. Architektūroje naudojami decentralizuoti identifikatoriai (DID), „Ethereum“ išmaniosios sutartys ir SHA-256 algoritmas DID generavimui ir patvirtinimui. Tai užtikrina asmeninės informacijos kontrolę per patvirtinamus įgaliojimus, saugią prieigą per privačius raktus ir HTTPS.

15 lentelė. **SST architektūros ir SST principų [CA16] palyginimas su sistemos reikalavimais**

Principai	SST architektūra	Aprašymas	Palyginimas	FR	NFR
Egzistavimas	Naudoja decentralizuotus identifikatorius (DID) nuolatiniam skaitmeniniam buvimui nustatyti. Vartotojai gali susikurti skaitmeninę piniginę, kurioje saugo savo DID. DID įgyvendinami kaip „Ethereum“ išmaniosios sutartys naudojant „Solidity“.	Pabrėžia asmenų egzistavimo skaitmeniniame pasaulyje svarbą.	Architektūra atitinka šį principą, nes asmenims suteikiama nuolatinė skaitmeninė buvimo vieta naudojant DID ir skaitmenines pinigines. DID įgyvendinami kaip „Ethereum“ išmaniosios sutartys naudojant „Solidity“, o SHA-256 šifravimo algoritmas užtikrina DID vientisumą ir saugumą.	FR.1 FR.2 FR.3 FR.4	NFR.1
Valdymas	Leidžia asmenims išlaikyti asmeninės informacijos kontrolę, naudojant patvirtinamus įgaliojimus. Vartotojai gali pridėti, redaguoti ir ištrinti skaitmeninės piniginės duomenis. Patvirtinami įgaliojimai pateikiami kaip „Ethereum“ išmaniosios sutartys naudojant „Solidity“.	Pasisako už tai, kad asmenys turėtų teisę valdyti ir kontroliuoti savo tapatybę.	Architektūra atitinka šį principą, nes asmenims suteikiama asmeninės informacijos kontrolė ir jie gali ją valdyti savo skaitmeninėje piniginėje. Patvirtinami įgaliojimai pateikiami kaip „Ethereum“ išmaniosios sutartys, taip užtikrinant duomenų vientisumą ir nekintamumą.	FR.11 FR.12 FR.13	NFR.8
Prieiga	Suteikiama saugi ir patogi prieiga prie skaitmeninių tapatybių. Naudotojai gali saugiai pasiekti ir valdyti savo skaitmeninę piniginę	Pripažįsta, kad svarbu, jog asmenys turėtų lengvą prieigą prie savo informacijos.	Architektūra atitinka šį principą užtikrindama saugią ir patogią prieigą prie skaitmeninių tapatybių naudojant privačius raktus ir prieigą prie	FR.2, FR.11	NFR.2

Principai	SST architektūra	Aprašymas	Palyginimas	FR	NFR
	naudodami privatų raktą. Saugiam bendravimui naudojamas HTTPS protokolas.		skaitmeninės piniginės. HTTPS protokolas užtikrina skaitmeninės piniginės ir DID agento ryšio konfidencialumą ir vientisumą.		
Skaidrumas	Pasitelkiama išskirstytos apskaitos technologija skaidriam ir audituojamam įgaliojimų valdymui. Į blokų grandinę įrašomi patvirtinami įgaliojimai ir sandoriai.	Pabrėžia skaidrumo svarbą kuriant pasitikėjimą.	Architektūra atitinka šį principą, nes naudojama išskirstytos apskaitos technologija, užtikrinanti skaidrumą ir audituojamą įgaliojimų valdymą. Patvirtinami įgaliojimai ir sandoriai įrašomi į Ethereum blokų grandinę, taip užtikrinant duomenų skaidrumą ir nekintamumą.	FR.15	-
Patvarumas	Naudoja išskirstytos apskaitos technologiją, kad užtikrintų skaitmeninių tapatybių patvarumą ir prieinamumą. Skaitmeninės piniginės duomenys saugomi prietaise ir gali būti atsarginės kopijos arba atkurti.	Pabrėžia atkaklumo svarbą išlaikant asmenų buvimą.	Architektūra atitinka šį principą, nes, siekiant užtikrinti skaitmeninių tapatybių ir piniginių duomenų patvarumą ir prieinamumą, naudojama išskirstytos apskaitos technologiją. Skaitmeninės piniginės duomenys saugomi naudotojo įrenginyje ir gali būti sukurtos jų atsarginės kopijos arba jie gali būti atkurti, kad būtų išlaikytas prieinamumas ir tęstinumas.	FR.11, FR.12, FR.14	NFR.3
Perkeliamumas	Palaiko sąveikius standartus ir protokolus, kad būtų užtikrintas sklandus skaitmeninių tapatybių perkeliamumas. Laikosi W3C patvirtinamų įgaliojimų ir Decentralizuotų identifikatorių standartų.	Pripažįsta skaitmeninių tapatybių naudojimo įvairiuose kontekstuose svarbą.	Architektūra palaiko perkeliamumą pasitelkiant sąveikius standartus, pavyzdžiui, W3C patvirtinamus įgaliojimus ir decentralizuotus identifikatorius (DID), leidžiančius saugiai ir veiksmingai perduoti skaitmenines tapatybes įvairiose sistemose ir platformose.	FR.6 FR.9 FR.10 FR.22-24 FR.26-33	–

7. W3C decentralizuotų identifikatorių specifikacijos problemos ir taisymo rekomendacijos

Lentelėje nr. 16 įtrauktos pažeidžiamosios vietos, susijusios su savarankiškos suverenios tapatybės sistemomis, kartu su siūlomais sprendimais ir jų atitikimu 15 lentelėje nustatytiems principams. Pažeidžiamumai apima įvairius aspektus, pavyzdžiui, nepakankamą įvesties patvirtinimą, netinkamą privačių raktų apsaugą, kriptografijos pažeidžiamumą, autorizacijos kontrolės trūkumus, pakartotinių atakų pažeidžiamumą, patvirtinimo ir patikrinimo trūkumus, saugių atsarginių kopijų ir atkūrimo mechanizmų trūkumą, polinkį į "man-in-the-middle" atakas, saugaus kodavimo praktikos neįgyvendinimą ir netinkamą DID dokumentų saugojimo apsaugą. Siūlomuose sprendimuose siūlomos priemonės šioms pažeidžiamoms vietoms pašalinti, pabrėžiant griežto įvesties patvirtinimo, patikimo raktų valdymo, kriptografijos standartų laikymosi, prieigos kontrolės įgyvendinimo, saugių ryšių protokolų naudojimo, DID metodų ir formatų patvirtinimo, saugių atsarginių kopijų darymo procedūrų, saugaus kodavimo praktikos naudojimo ir DID dokumentų šifravimo svarbą.

16 lentelė. **W3C decentralizuotų identifikatorių specifikacijos papildymo rekomendacijos**

Nr	Problema	Siūlomas sprendimas	SST principas
1	Tinkamo įvesties patvirtinimo trūkumas	Realizuoti griežtus įvesties patvirtinimo mechanizmus, kad išvengti kenkėjiškos įvesties ir išvalyti duomenis, kad išvengti įskiepijimo atakų.	Valdymas
2	Netinkama privačių raktų apsauga	Naudoti patikimą raktų valdymo praktiką, įskaitant saugų raktų saugojimą, šifravimą ir saugius raktų generavimo algoritmus.	Valdymas
3	Nepakankama apsauga nuo kriptografinių pažeidžiamumų	Atnaujinti naujausius kriptografijos standartus.	Valdymas, Skaidrumas
4	Tinkamos autorizavimo kontrolės nebuvimas	Įdiegti prieigos kontrolės mechanizmus, kuriais užtikrinamas tinkamas DID operacijų autorizavimas ir privilegijų lygiai.	Valdymas
5	Nepakankama apsauga nuo pakartojimo atakų	Naudoti tokius mechanizmus, kaip nonce arba laiko žyma pagrįsta patikra, kad išvengti pakartotinių atakų ir užtikrinti pranešimų šviežumą.	Valdymas
6	Nepakankamas patvirtinamų įgaliojimų patvirtinimas ir patikrinimas	Įdiegti patikimus patvirtinimo ir tikrinimo procesus tikrintiniems įgaliojimams, įskaitant parašo tikrinimą ir vientisumo patikras.	Valdymas, Prieiga
7	Saugių atsarginių kopijų kūrimo ir atkūrimo mechanizmų trūkumas	Įdiegti saugias svarbiausių duomenų atsarginių kopijų kūrimo procedūras, įskaitant šifruotas atsargines kopijas, ir sukurti patikimus atkūrimo procesus.	Valdymas, Skaidrumas

Nr	Problema	Siūlomas sprendimas	SST principas
8	Nepakankama apsauga nuo "man-in-the-middle" atakų	Naudoti saugius ryšio kanalus, pvz., TLS arba HTTPS, kad būtų išvengta pasiklausymo ar klastojimo DID ryšio metu.	Patvarumas, Valdymas
9	DID dokumentų saugojimo trūkumas	Užšifruoti skaitmeninėse piniginėse ar saugyklose saugomus DID dokumentus, kad išvengti neteisėtos prieigos ar klastojimo.	Valdymas
10	Nepakankama ryšių tarp DID agentų apsauga	Įgyvendinti saugius ryšių protokolus, pavyzdžiui, abipusį autentiškumo patvirtinimą ir pranešimų šifravimą, kad būtų užtikrintas ryšių kanalų konfidencialumas ir vientisumas.	Valdymas
11	Nepakankamas DID metodų ir formatų patvirtinimas	Užtikrinti, kad būtų laikomasi tinkamų DID metodų ir formatų, kad būtų išvengta negaliojančių ar nestandartinių DID priėmimo.	Patvarumas
12	Nepakankama DID atkūrimo mechanizmų apsauga	Realizuoti saugias DID atkūrimo procedūras, pavyzdžiui, naudoti saugius paskyros atkūrimo procesus, daugiafaktorinį autentiškumo patvirtinimą ir saugias atsargines kopijas bei atkūrimo mechanizmus.	Valdymas

REZULTATAI IR IŠVADOS

Rezultatai:

1. Identifikuoti SST sistemos naudojimo atvejai.
2. Identifikuoti funkciniai ir nefunkciniai SST sistemos reikalavimai.
3. Sukurta SST sistemos architektūra pagal W3C decentralizuotų identifikatorių specifikaciją.
4. Sukurtas SST sistemos prototipas pagal sukurtą SST sistemos architektūrą.
5. Patikrinta sukurta SST sistemos architektūra ir reikalavimai, remiantis Kristoferio Aleno pasiūlytais SST technologijos principais.
6. Identifikuotos W3C decentralizuotų identifikatorių specifikacijos problemos ir pasiūlytos taisymo rekomendacijos.

Išvados:

1. Tikslių standartų kaip turėtų būti realizuojama savarankiška suvereni tapatybė dar nėra.
2. Kontekstiniai failai minimi W3C specifikacijose [RSLA19, SLC19], nepalaiko visų laukų, kurie yra minimi pačiose specifikacijose.

SĄVOKŲ APIBRĖŽIMAI

- DID (angl. decentralized identifier) – decentralizuotas identifikatorius.
- URI (angl. uniform resource identifier) – simbolių eilutė vienaraikšmiškai identifikuojanti resursą. Siekiant išsaugoti vienodumą, URIs turi iš anksto nustatytas sintaksės taisykles, tačiau taip pat palaiko hierarchinių pavadinimų schemą.
- URL (angl. uniform resource locator) – apibrėžiamas kaip žiniatinklio adresas turintisnuorodą į žiniatinklio šaltinį ir machanizmą, kaip nurodytą nuorodą gauti. URL yra URI tipas.
- VC (angl. verifiable credential) – realaus pasaulio įgaliojimų atitikmuo virtualioje erdvėje, turintis kriptografinį pagrindą.
- DLT (angl. distributed ledger technology) – sąvoka naudojama apibūdinti viešą duomenų bazę, kur kiekviena šalis ar pasirinkta grupė turi tą pačią duomenų bazės kopiją ir kiekviena šalis gali daryti pakeitimus apskaitoje, taip išlaikant viešą duomenų bazę, kur nėra šalies, kuriai priklauso pati duomenų bazė, nes ji yra bendra.
- SST (angl. self-sovereign identity) – skaitmeninės tapatybės tipas, kai visas administravimo ir kontrolės teises turi tapatybės savininkas.
- W3C (angl. World Wide Web Concorsium) – tarptautinė bendruomenė, kuri kuria viešus standartus ir specifikacijas interneto technologijoms, kad užtikrintų ilgalaikį interneto augimą.
- UML (angl. unified modeling language) – modeliavimo ir specifikacijų kūrimo kalba, skirta specifikuoti, atvaizduoti ir konstruoti objektiškai orientuotų programų dokumentus.
- API (angl. application programming interface) – ryšio protokolų ir programinės įrangos kūrimo priemonių rinkinys. API palengvina kompiuterinių programų kūrimą.
- SSO (angl. single sign-on) – programinės įrangos sistema, suteikianti galimybę naudotojams naudotis internetinės tapatybės teikėju registruojantis ir jungiantis prie internetinių aplikacijų, siekiant išvengti skirtingų skaitmeninių tapatybių kūrimo kiekvienoje iš jų.

ŠALTINIAI

[AMS18] Angel Angelov, Mihail Milkov, Markus Sørensen. *Decentralized Identity Management System for Self-Soverign Identity.* 2018. (p. 35-36) [žiūrėta 2022 m. Rugsėjo 6 d.] Prieiga per internetą: (https://projekter.aau.dk/projekter/files/281068659/Master_Thesis_ICTE4SER4.2.pdf).

[CA16] Christopher Allen. *The Path to Self-Sovereign Identity.* 2016. [žiūrėta 2022 m. rugsėjo 6 d.] Prieiga per internetą: (http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity.html).

[DB16] Djuri Baars. *Towards Self Sovereign Identity using Blockchain Technology.* 2016. (p. 1–2) [žiūrėta 2022 m. rugsėjo 6 d.] Prieiga per internetą: (https://essay.utwente.nl/71274/1/Baars_MA_BMS.pdf).

[ITRC22] Identity Theft Resource Center. Identity Theft Resource Center's 2021 Annual Data Breach Report Sets New Record for Number of Compromises. 2022. [žiūrėta 2023 m. gegužės 6 d.] Prieiga per internetą: (https://www.idtheftcenter.org/post/identity-theft-resource-center-2021-annual-data-breach-report-sets-new-record-for-number-of-compromises/).

[JA16] Joe Andrieu. *A technology free definition of self-sovereign identity.* 2016. [žiūrėta 2022 m. vasarios 9 d.] Prieiga per internetą: (https://raw.githubusercontent.com/jandrieu/rebooting-the-web-of-trust-fall2016/master/topics-and-advance-readings/a-technology-free-definition-of-self-sovereign-identity.pdf).

[LA20] Allende Lopez, Marcos. Self-Sovereign Identity: The Future of Identity: Self-Sovereignity, Digital Wallets, and Blockchain. 2020. [žiūrėta 2023 m. gegužės 5 d.]

[RSLA19] Drummond Reed, Manu Sporny, Dave Longley, Christopher Allen. *Decentralized Identifiers (DIDs)*. 2019. [žiūrėta 2022 m. rugsėjo 6 d.] Prieiga per internetą: (https://w3c-ccg.github.io/did-spec/).

[SF17] Sovrin Foundation. *The Inevitable Rise of Self-Sovereign Identity*. 2017. [žiūrėta 2022 m. rugsėjo 6 d.] Prieiga per internetą: (https://sovrin.org/wp-content/uploads/2017/06/The-Inevitable-Rise-of-Self-Sovereign-Identity.pdf).

[SG18] Swati Goyal. *History of Blockchain Technology – Timeline Infographic.* 2018. [žiūrėta 2022 m. rugsėjo 6 d.] Prieiga per internetą: (https://101blockchains.com/h

istory-of-blockchain-timeline/).

[SLC19] Manu Sporny, Dave Longley, David Chadwick. *Verifiable Credentials Data Model 1.0.* 2019. [žiūrėta 2022 m. rugsėjo 6 d.] Prieiga per internetą: (https://www.w3.org/TR/verifiable-claims-data-model/).

[SLKL18] Manu Sporny, Dave Longley, Gregg Kellogg, Markus Lanthaler. *JSON-LD 1.1.* 2018. [žiūrėta 2022 m. rugsėjo 6 d.] Prieiga per internetą: (https://www.w3.org/2018/jsonld-cg-reports/json-ld/).

[OLSBSE19] Nate Otto, Sunny Lee, Brian Sletten, Daniel Burnett, Manu Sporny, Ken Ebert. *Verifiable Credentials Use Cases.* 2019. [žiūrėta 2023 m. sausio 19 d.] Prieiga per internetą: (https://www.w3.org/TR/vc-use-cases/).

PRIEDAI

1. priedas. SDK bibliotekos dokumentacija

DID kūrimas:

Aprašymas: DID generavimas pagal pasirinktą metodą.

```
import * as DidUtils from '@vsaulis-did/utils/did'

const method = "defined";
const did = DidUtils.generateDid();
// did = did:defined:d8c75b9e-d62d-11ec-9d64-0242ac120002
```

DID tikrinimas:

Aprašymas: DID tikrinimas pagal reguliariąją išraišką.

```
import * as DidUtils from '@vsaulis-did/utils/did'

const did = "did:defined:d8c75b9e-d62d-11ec-9d64-0242ac120002";
const { status } = DidUtils.validate(did);
// status = "success"
```

```
import * as DidUtils from '@vsaulis-did/utils/did'

const did = "id:defined:d8c75b9e-d62d-11ec-9d64-0242ac120002";
const { status, message } = DidUtils.validate(did);
// status = "error"
// message = "DID is invalid"
```

Viešo ir privataus raktų poros generavimas:

Aprašymas: Viešo ir privataus rakto generavimas pasirinktu formatu.

```
import * as KeysUtils from '@vsaulis-did/utils/keys'

const format = "rsa";
const { publicKey, privateKey } = KeysUtils.generateKeyPair(format)
// publicKey = "-----BEGIN RSA PUBLIC KEY-----MIIBCgKCAQEA+xGZ..."
// privateKey = "-----BEGIN RSA PRIVATE KEY-----9mxDXDf6AU0cN/..."
```

DID dokumento kūrimas:

Aprašymas: DID dokumento generavimas pagal pasirinktą DID ir viešą raktą.

```
import * as DidDocumentUtils from '@vsaulis-did/utils/did-document'

const did = "did:defined:d8c75b9e-d62d-11ec-9d64-0242ac120002";
const publicKey = "-----BEGIN RSA PUBLIC KEY-----MIIBCgKCAQEA+xGZ...";
const didDocument = DidDocumentUtils.generateDidDocument({ did, publicKey })
```

DID dokumento pasirašymas:

Aprašymas: DID dokumento pasirašymas pagal viešo rakto DID.

```
import * as DidDocumentUtils from '@vsaulis-did/utils/did-document'

const did = "did:defined:d8c75b9e-d62d-11ec-9d64-0242ac120002";
const publicKey = "-----BEGIN RSA PUBLIC KEY-----MIIBCgKCAQEA+xGZ...";
const didDocument = DidDocumentUtils.generateDidDocument({ did, publicKey })

const publicKeyId = "did:defined:166evUDhpSSq23xP26oeQ4uNZNZce1KFFP";
const signature = KeysUtils.sign(publicKeyId, didDocument);
```

DID dokumento gavimas:

Aprašymas: DID dokumento gavimas pagal DID.

```
import * as DidDocumentUtils from '@vsaulis-did/utils/did-document'

const did = "did:defined:d8c75b9e-d62d-11ec-9d64-0242ac120002";
const { status, didDocument } = await DidDocumentUtils.findDidDocument(did)
// status = "success"
// didDocument = {"@context": ["https://w3id.org/did/v1"], ...}
```

```
import * as DidDocumentUtils from '@vsaulis-did/utils/did-document'

const did = "did:defined:d8c75b9e-d62d-11ec-9d64-0242ac120005";
const { status, didDocument } = await DidDocumentUtils.findDidDocument(did)
// status = "notFound"
// didDocument = null
```

Patvirtinamo įgaliojimo sukūrimas:

Aprašymas: Patvirtinamo įgaliojimo sukūrimas patvirtinti išsilavinimą pagal DID.

```
import * as VerifiableCredentialsUtils from '@vsaulis-did/utils/verifiable-credentials';
import * as KeysUtils from '@vsaulis-did/utils/keys';

const did = "did:defined:d8c75b9e-d62d-11ec-9d64-0242ac120002";
const expirationDate = Date.now();
const type: CredentialType = "UniversityDegreeCredential";
const credentialSubject = { id: did, degree: "BachelorDegree" };

const veriafiableCredential = await VerifiableCredentialsUtils.createVerifiableCredential({
  did,
  expirationDate,
  type,
  credentialSubject
});
```

Patvirtinamo įgaliojimo pasirašymas:

Aprašymas: Patvirtinamo įgaliojimo pasirašymas pagal viešo rakto DID.

```
import * as VerifiableCredentialsUtils from '@vsaulis-did/utils/verifiable-credentials';
import * as KeysUtils from '@vsaulis-did/utils/keys';

const did = "did:defined:d8c75b9e-d62d-11ec-9d64-0242ac120002";
const expirationDate = Date.now();
const type: CredentialType = "UniversityDegreeCredential";
const credentialSubject = { id: did, degree: "BachelorDegree" };

const veriafiableCredential = await VerifiableCredentialsUtils.createVerifiableCredential({
  did,
  expirationDate,
  type,
  credentialSubject
});

const publicKeyId = "did:defined:166evUDhpSSq23xP26oeQ4uNZNZce1KFFP";
const signature = KeysUtils.sign(publicKeyId, veriafiableCredential);
```

DID dokumento parašo patikrinimas:

Aprašymas: DID dokumento patikrinimas ar jis pasirašytas nurodytu viešu raktu.

```
import * as DidDocumentUtils from '@vsaulis-did/utils/did-document'

const publicKey = "-----BEGIN RSA PUBLIC KEY-----MIIBCgKCAQEA+xGZ...";

const didDocument = {
  '@context': ['https://w3id.org/did/v1', 'https://w3id.org/security/v1'],
  id: 'did:defined:166evUDhpSSq23xP26oeQ4uNZNZce1KFFP',
  publicKey: [
    {
      id: 'did:defined:166evUDhpSSq23xP26oeQ4uNZNZce1KFFP',
      owner: 'did:defined:166evUDhpSSq23xP26oeQ4uNZNZce1KFFP',
      type: 'Ed25519VerificationKey2018',
      publicKeyHex: '0023757068c19dd7b51531ed3',
    },
    {
      id: 'did:defined:166evUDhpSSq23xP26oeQ4uNZNZce1KFFP',
      owner: 'did:defined:166evUDhpSSq23xP26oeQ4uNZNZce1KFFP',
      type: 'Ed25519VerificationKey2018',
      publicKeyHex: '00e8b5ce1df5c8984511591967b17ae0',
    },
  ],
  service: [
    {
      serviceName: 'openid',
      type: 'OpenIdConnectVersion1.0Service',
      serviceEndpoint: 'https://openid.example.com/',
    },
    {
      serviceName: 'vcr',
      type: 'CredentialRepositoryService',
      serviceEndpoint: 'https://repository.example.com/service/8377464',
    },
    {
      serviceName: 'xdi',
      type: 'XdiService',
      serviceEndpoint: 'https://xdi.example.com/8377464',
    },
  ],
  proof: {
    type: 'LinkedDataSignature2015',
    created: '2016-02-08T16:02:20Z',
    creator: 'did:example:8uQhQMGzWxR8vw5P3UWH1ja#keys-1',
    signatureValue: 'QNB13Y7Q9...1tzjn4w==',
  },
  created: '2019-02-11T07:27:43.687Z',
  updated: '2019-02-11T07:27:43.687Z',
};

const status = await DidDocumentUtils.checkSignature(didDocument, publicKey)
// status = "valid"
```