VILNIAUS UNIVERSITETO

KAUNO FAKULTETAS

SOCIALINIŲ MOKSLŲ IR TAIKOMOSIOS INFORMATIKOS INSTITUTAS

Finansų technologijų studijų programa

Kodas 6211BX022

MINDAUGAS DUOBLYS

MAGISTRO BAIGIAMASIS DARBAS

DECENTRALIZUOTŲ SANDORIŲ APSAUGOS SISTEMA TAIKANT PAKEITIMO MOKESČIU METODĄ

Kaunas 2025

VILNIAUS UNIVERSITETO
KAUNO FAKULTETAS
SOCIALINIŲ MOKSLŲ IR TAIKOMOSIOS INFORMATIKOS INSTITUTAS

MINDAUGAS DUOBLYS

MAGISTRO BAIGIAMASIS DARBAS

DECENTRALIZUOTŲ SANDORIŲ APSAUGOS SISTEMA TAIKANT PAKEITIMO MOKESČIU METODĄ

Magistrantas _____
(parašas)

Darbo vadovas _____
(parašas)

_____
(darbo vadovo mokslo laipsnis, mokslo pedagoginis vardas, vardas ir pavardė)

Darbo įteikimo data
Registracijos Nr.

Kaunas 2025

TURINYS

SANTRUMPŲ SĄRAŠAS

ERC20 – „Ethereum“ tinklo žetonų standartas

ETH – eteris (angl. *ethereum*), „Ethereum“ tinklo valiuta

EVM – Ethereum virtuali mašina

IRT – informacinės ryšių technologijos

MBD25 – darbe sukurto ERC20 žetono simbolis

MEV – maksimaliai išgaunama vertė (angl. maximal extractable value)

MiCA – Europos Parlemento ir Tarybos reglamentas dėl kriptoturto rinkų (angl. Markets in Crypto-Assets)

WETH – „įvyniotas“ eteris (angl. *wrapped ethereum*), „Ethereum“ tinklo valiuta, paversta į ERC20 tipo žetoną

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

Duoblys, Mindaugas (2025). *A Decentralized Transaction Protection System Using the Replace-By-Fee Method.* MBA Graduation Paper. Kaunas: Vilnius University, Kaunas Faculty, Institute of Social Sciences and Applied Informatics. 97 p.

SUMMARY

Automated trading is a form of trading that relies on automated systems which execute numerous transactions in exchanges in a very short – in milisecords or seconds – period of time to earn profit. The main goal of this strategy is to exploit short-term price fluctuations in order to take an advantage of (un)suitable market situation. However, the strategy of algorithmic trading not only exploits existing price differences, but, how Jones (2013) notices, also uses available information manipulate the market to exploit other market participants who become victims. Such actions are contrary to the principles of fairness and fall under the definition of manipulation. According to the regulation of the European Parliament and the Council (2023), actions that are false or misleading in terms of demand or supply, or maintaining prices at an artificial level, are classified as market manipulation, which directly corresponds to the mechanics of frontrunning attacks in decentralized systems, where prices are artificially distorted.

Blockchain technology is unique so that every information about past and ongoing transactions are publicly available, unchanging and decentralised. Gomez-Trujillo, Gonzalez-Perez ir Velez-Ocampo (2021) emphasizes this operational principle allows to execute real-time market analysis to assess how trades are done and to find suspicious trading patterns. In this context, the technical feature of replace-by-fee plays a key role, which, according to Narayanan, Bonneau, Felten, Miller and Goldfeder (2016), allows uncofirmed transactions to be replaced with new ones. This means that status of transactions prior to inclusion in a block is not final, and the order of transactions is technically adjustable.

This intersection of automated trading strategies, blockchain characteristics and legal requirements creates conditions for the emergence of preventive measures – an automated protection system based on monitoring frontrunning attacks on the network and replace-by-fee mechanism that allows threats to be responded to at the same speed as market manipulation.

This thesis consists of 97 pages, 10 tables and 7 figures.

ĮVADAS

Automatizuota prekyba yra prekybos forma, kuri paremta automatizuotomis sistemomis, kurios per labai trumpą – milisekundžių ar sekundžių - laiką atlieka itin daug prekybos veiksmų vertybinių popierių biržose, siekiant uždirbti pelną. Pagrindinė šios strategijos esmė – išnaudoti trumpalaikius kainų svyravimus, kad būtų pasinaudota (ne)tinkama rinkos situacija. Algoritminė prekyba ne tik išnaudoja jau egzistuojančius kainų skirtumus, kaip tai pastebi Jones (2013), tačiau ir naudojasi prieinama informacija blokų grandinėje, jog būtų manipuliuojama rinka ir išnaudojami kiti rinkos dalyviai, kurie tampa aukomis. Tokie veiksmai neatitinka sąžiningumo principų ir patenka į manipuliacijos apibrėžimą. Remiantis Europos Parlamento ir Tarybos reglamentu dėl kriptoturto rinkų (2023), veiksmai, kurie yra netikri ar klaidinantys dėl paklausos ar pasiūlos arba kainos palaikymas dirbtiniame lygmenyje, yra klasifikuojami kaip manipuliavimas rinka, o tai tiesiogiai atitinka priešakinės transakcijos (angl. *frontrunning*) atakų mechaniką decentralizuotose sistemose, kuomet dirbtinai iškreipiama kaina.

Blokų grandinės technologija yra išskirtinė tuo, kad visa informacija apie sudaromus sandorius yra viešai prieinama, ji nekinta bei yra decentralizuota. Gomez-Trujillo, Gonzalez-Perez ir Velez-Ocampo (2021) teigimu, dėl tokio veikimo principo yra sudaroma galimybė vykdyti rinkos analizę realiu laiku – vertinti, kaip vykdoma prekyba, ar ieškoti įtartinų veiksmų modelių. Šiame kontekste esminį vaidmenį atlieka techninė transakcijų pakeitimo mokesčiu (angl. *replace-by-fee*) savybė, kuri, anot Narayanan, Bonneau, Felten, Miller ir Goldfeder (2016), leidžia dar nepatvirtintas transakcijas pakeisti naujomis. Tai reiškia, kad sandorių būsena iki įtraukimo į bloką nėra galutinė, o transakcijų eiliškumas yra techniškai koreguojamas.

Ši technologinė automatizuotos prekybos strategijų ir blokų grandinių ypatumų bei teisinių reikalavimų sankirta sukuria sąlygas ir prevencinių priemonių atsiradimui – automatizuotai apsaugos sistemai, paremtai atakų stebėjimu tinkle ir „replace-by-fee“ mechanizmu, leidžiančiai reaguoti į grėsmes tuo pačiu metu, kuriuo veikia ir rinkos manipuliacijos.

Šį rašto darbą sudaro 97 puslapiai, 10 lentelių ir 7 paveikslai.

Temos aktualumas

Vienas iš blokų grandinės bruožų, kad tai yra sąžininga ir skaidri prekybos platforma, gali pasirodyti klaidingas, kuomet pradedama vertinti, kaip plačiai yra paplitęs ir vis didėja automatizuotų prekybos strategijų pritaikymas blokų grandinėje. Išnaudojant blokų grandinės viešumo principą, tokios automatizuotos prekybos sistemos, pasak Daian, Goldfeder, Kell, Li, Zhao, Benov, Breidenbach ir Juels (2019), naudojasi decentralizuotų keityklų neefektyvumu ir gali leisti mokėti didelius transakcijų

mokesčius, optimizuoti tinklo delsą, siekiant pasinaudoti įprastinių rinkos dalyvių sandoriais juos aplenkiant operacijų įvykdyme.

Šis technologinis piktnaudžiavimas nėra tik teorinė grėsmė. Remiantis Qin, Zhou ir Gervais (2021) atlikta analize, per 32 tyrimo mėnesius pavyko nustatyti maždaug 750 tūkstančių įvykdytų atakų, susijusių su algoritminės prekybos strategijomis, o jų žala rinkos dalyviams siekia apie 174 milijonus JAV dolerių. Tai rodo, kad priešakinės transakcijos atakos kasdien tiesiogiai nusavina lėšas iš nieko neįtariančių rinkos dalyvių.

Be tiesioginės finansinės žalos, priešakinių transakcijų atakų aktualumas išauga, kuomet vertinamas platesnis rinkos manipuliacijų spektras. Anot Zhou, Qin, Torres, Le bei Gervais (2020), ši ataka yra fundamentalus komponentas kitoms didelio dažnio prekybos atakoms. Tai reiškia, kad bandymai sustabdyti priešakinės transakcijos ataką reiškia ir kitų, pavyzdžiui, dvipusio įterpimo arba kitaip žinomos kaip „sumuštinio“ (angl. *sandwich*) atakų sustabdymą, o tai leidžia užkirsti ne tik pavienes, bet ir kompleksines atakas.

Apibendrinant galima teigti, kad automatizuota priešakinė ataka yra opi problema stambiu decentralizuotos rinkos mastu. Ji ne tik sukelia finansinius nuostolius, tačiau yra ir rinkos manipuliacijos įrankis.

Problemos ištyrimo lygis

Nors tradicinėse rinkose reguliavimas jau yra ganėtinai išplėtotas, decentralizuotose platformose vis dar trūksta efektyvių priemonių, galinčių apsaugoti nuo nesąžiningos prekybos ir manipuliacijos. Mokslinėje literatūroje šios rizikos dažniausiai nagrinėjamos teoriškai, pateikiant bendrinius tobulinimo siūlymus, bet, kaip pastebi S. Yang, Zhang, Huang, Chen, Y. Yang ir Zhu (2022), praktiniams sprendimams skiriama nepakankamai dėmesio.

Labiausiai aptariama algoritminė prekybos strategija, vadinama maksimaliai išgaunama verte arba MEV (angl. *maximal extractable value*), pasireiškia transakcijų tvarkos manipuliacija bloke. Bartoletti ir Zunino (2023) formalizuodami šią strategiją, kaip problemą blokų grandinei, linksta prie išvadų, kad strategija sukelia rinkos iškraipymus ir pažeidžia rinkos dalyvių interesus.

Siekiant kovoti su rinkos manipuliacija, buvo sukurta sistema „Flashbots“, skirta mažinti transakcijų perrikiavimą, būdingą automatizuotoms prekybos strategijoms. Kaip pastebi Weintraub, Torres, Nita-Rotaru ir State (2022), nors tai ir sumažino transakcijų perrikiavimo problemą, sistemos naudotojai, perimdami beveik visą skaičiavimo rinkos dalį, tik padidino gaunamo pelno nelygybę.

Darbo problema

Kaip efektyviai transakcijų pakeitimo mokesčiu metodo taikymas blokų grandinėje padėtų sumažinti automatizuotų atakų žalą?

Darbo objektas

Automatizuoti sandoriai ir manipuliacijų prevencija blokų grandinėje

Darbo tikslas

Įvertinti išmaniųjų sutarčių ir transakcijų pakeitimo mokesčiu mechanizmo taikymo efektyvumą decentralizuotoje prekyboje, siekiant sumažinti automatizuotų atakų riziką.

Darbo uždaviniai

1. Išanalizuoti blokų grandinės ir decentralizuotų finansų veikimo principus bei išmaniųjų kontraktų galimybes kuriant automatizuotus prevencijos mechanizmus.
2. Formalizuoti automatizuotos prekybos sistemas ir būdingas nesąžiningos prekybos savybes.
3. Pasiūlyti apsaugos metodą, skirtą identifikuoti galimus nesąžiningos prekybos atvejus ir neutralizuoti atakas, taikant transakcijų pakeitimu mokesčiu.
4. Remiantis pasiūlytu metodu, suformuluoti automatizuotų strategijų atakų apsaugos sistemos reikalavimus bei sukurti veikiantį prototipą.
5. Įvertinti pasiūlyto metodo efektyvumą, privalumus ir apribojimus.

Darbo struktūra

Pirmojoje – literatūros analizės – dalyje yra analizuojami blokų grandinės ypatybės ir techninės išmaniųjų kontraktų savybės, formalizuojami automatizuotos prekybos ypatumai ir apibrėžiamos nesąžiningos prekybos strategijos bei taikomi apsaugos mechanizmai.

Antrojoje – siūlo sprendimo – darbo dalyje nagrinėjami siūlomos sistemos komponentų reikšmingumas ir vertinimas tokio sprendimo poreikis, remiantis reguliacinėmis įžvalgomis. Pateikiami apsaugos sistemos vertinimo kriterijai.

Trečiojoje – eksperimentinėje - dalyje realizuojama pasiūlyta apsaugos nuo atakų sistema, nustatomi testavimo scenarijai ir jie vykdomi, atliekamas rezultatų aptarimas.

Darbo pabaigoje pateikiamos siūlomos apsaugos sistemos išvados, apimančios atliktą analizę, siūlomą sprendimo pasiūlymą bei įgyvendintą sistemą, pateikiami tolimesni pasiūlymai.

Literatūros šaltiniai

Teorinėje dalyje analizuojama užsienio autorių mokslinė literatūra, apimanti blokų grandinę, išmaniųjų kontraktų ir automatizuotos prekybos aspektus. Praktinė dalis remiasi naudotų technologijų ir paslaugų tiekėjų dokumentacijomis.

Darbo ir tyrimo metodai

Teorinė blokų grandinės, išmaniųjų kontraktų ir automatizuotos prekybos analizės dalis atlikta pasinaudojant mokslinės literatūros analize, sinteze, indukcija, klasifikavimu ir lyginamąja analize. Siūlomam sprendimui pagrįsti pritaikyta teisinių dokumentų bei lyginamoji technologijų analizė, architektūra apibrėžta taikant sistemų modeliavimą. Eksperimentinėje dalyje surinktiems duomenims apdoroti naudota kiekybinė, lyginamoji analizė, aprašomoji statistika.

Darbo teorinė reikšmė

- Atskleistas blokų grandinės skaidrumo ir viešumo principo pažeidžiamumas decentralizuotų sandorių saugumo kontekste
- Identifikuotos išmaniųjų kontraktų techninės realaus laiko rinkos manipuliacijų prevencijos architektūrinės ribos ir taikymo galimybės
- Susistemintos automatizuotų prekybos strategijos ir rinkos manipuliacijų skiriamieji požymiai

Darbo praktinė reikšmė

- Išanalizavus automatizuotos prekybos strategijas ir blokų grandinės ypatybes, pasiūlyta apsaugos nuo atakų sistema, apjungianti tinklo stebėjimą už grandinės ribų ir vykdymo mechanizmą blokų grandinėje.
- Pritaikytas jau blokų grandinėje egzistuojantis transakcijų pakeitimo mokesčiu mechanizmas gynybai prieš realiu laiku vykstančias atakas.

Darbo apribojimai ir sunkumai

- Mokslinėje literatūroje transakcijų pakeitimas mokesčiu tik minimas kaip blokų grandinės technologijos elementas, tačiau tikslesnės veikimo detalės nėra aptariamos.
- Tyrimas atliekamas „Sepolia“ testiniame tinkle, kuriame sudėtinga pilnai simuliuoti realaus tinklo elgseną.
- Prototipo testavimas vykdytas naudojant viešą „Infura“ paslaugą, kuri pasižymi didesniu vėlavimu.
- Testiniame tinkle naudojamų testinių žetonų ištekliai, prieinami per viešas dalinimo sistemas, vadinamas „kranais“ (angl. *faucets*) yra labai riboti, todėl sudėtinga simuliuoti tyrimo scenarijus, kuriuose reikalingas didelis žetonų kiekis.

Darbo struktūra ir apimtis

Rašto darbas yra sudarytas iš įvado, trijų skyrių, išvadų bei pasiūlymų. Pagrindinė darbo dalis apima 53 puslapius, 7 paveikslus ir 10 lentelių. Taip pat yra pateikiami 6 priedai, o literatūros sąraše nurodyti 34 šaltiniai.

1. DECENTRALIZUOTŲ FINANSŲ PRINCIPŲ TAIKYMO ANALIZĖ PAŽEIDIMŲ NUSTATYMUI

Šioje dalyje pateikiami blokų grandinės ir išmaniųjų kontraktų principai bei aptariamos automatizuotos prekybos strategijos, jų įtaką rinkos skaidrumui, rizikos ir galimybės pažeidimų identifikavimui ir prevencijai.

1.1. Blokų grandinės technologijos ir skaidrumo užtikrinimo principai

Blokų grandinės technologijos pradžia gali būti laikomas momentas, kuomet buvo paskelbtas straipsnis, aprašantis tarpusavio elektroninių pinigų sistemą, pasinaudojant kriptografija. Nakamoto (2008) pristatyta virtualios valiutos – *Bitcoin* – idėja atvėrė kelią į blokų grandine grįstas mokėjimo sistemas. Prasidėjusi nauja era finansų technologijose suteikė erdvę tolimesniam vystymuisi ir blokų grandinės technologijų diegimui įvairiuose srityse, įskaitant ir automatizuotą prekybą.

Remiantis Zheng, Xie, Dai, Chen ir Wang (2017), šiuolaikinės blokų grandinių sistemos gali būti išskiriamos į tris kategorijas: viešąją, privačią ir konsorciumo. Šios blokų grandinių sistemų grupės skiriasi savo prieinamumu. Viešosiose sistemose visi įrašai yra prieinami visiems asmenims, o tai suteikia galimybę dalyvauti konsensuso procesuose. Tokiu atveju tokia blokų grandinė yra decentralizuota, o priešingu atveju privačios blokų grandinės yra centralizuotos, kadangi jas valdo viena organizacija. Jei blokų grandinę valdo kelios atrinktos organizacijos, tada ją būtų galima priskirti konsorciumo grupei. Esant šiai taksonomijai, galima išskirti pagrindinius – decentralizuotumo, nekintamumo, efektyvumo, viešumo – lyginamuosius skirtumus, kurie padeda atskleisti pačių blokų grandinių principus.

Decentralizuotumas. Blokų grandinė yra vienas iš duomenų bazių modelių, skirtų įrašyti informaciją apie vykdomas transakcijas su kopijomis visuose kompiuteriuose, esančiuose ir dalyvaujančiuose tinkle. Tokį apibrėžimą pateikia Gomez-Trujillo, Gonzalez-Perez ir Velez-Ocampo (2021). Tokia sistema gali talpinti duomenis nepriklausomu ir decentralizuotu būdu, kas leidžia informacijos saugojimą perkelti į daugiau nei vieno asmens atsakomybę, jog būtų užtikrintas saugumas. Kadangi įrašai, patekę į tinklą, yra verifikuojami plataus tinklo, decentralizuota sistema savaime gali atmesti piktavališkus bandymus manipuliuoti visa sistema.

Taigi, apibūdinus blokų grandinę kaip duomenų bazės modelį, galima teigti, jog vienas iš principų yra decentralizuotumas, kuris dėl savo prigimties sukuria pasitikėjimą visame naudotojų tinkle.

Skaidrumas ir privatumas. Skaidrumo ir privatumo palyginimo, pateikto Sas ir Khairuddin (2017), tarp blokų grandinės ir tradicinių bankinių sistemų yra tas, kad informacijos prieinamumą pastarajame variante riboja susijusios šalys, sudariusios sandorį, ir bankas. Privatumas blokų grandinėje

yra didesnis, kadangi nėra reikalaujama jokia asmeninė informacija, jog būtų galima dalyvauti tinkle – pavyzdžiui, vykdyti transakcijas Bitcoin valiuta. Skaidrumas, kuris yra vienas pagrindinių blokų grandinės principų, įgyvendinamas visų transakcijų ir piniginių viešu paskelbimu tinkle. Taip suteikiama galimybė bet kuriam naudotojui vertinti transakcijų autentiškumą ir vientisumą. Šiuo atžvilgiu galima teigti, jog blokų grandinė yra pseudoanoniminė: nors visų tinkle esančių piniginių adresai ir transakcijos yra viešos, sandorių (operacijų) sudarytojų šalims tenka atsakomybė, jog tapatybės niekados nebus susietos. Augant kriptovaliutų populiarumui, jei asmuo nori įgyti savų Bitcoin nesinaudojant kasimo procesu, neretu atveju pirmas žingsnis, anot Bergman ir Rajput (2021), yra keliauti į biržą. Vis didėjanti reguliacinė aplinka kriptovaliutoms, verčia tarpininkus vadovautis pinigų plovimo prevencijos įstatymais. Tai verčia rinkti informaciją apie asmenis, besinaudojančius tokiomis biržomis, ir tinkamai nustatyti tapatybę. Dėl šios priežasties didėja tikimybė, jog konkrečios piniginės gali būti susietos su konkrečiais asmenimis ir blokų grandinių suteikiamas privatumas mažėja. Kitas būdas susieti tapatybes su piniginėmis ir transakcijomis, tai įvairių paslaugų ir prekių tiekėjų, priimančių mokėjimus bitcoin valiuta, informacijos rinkimas, siekiant vykdyti užsakymus. Dažnu atveju surenkama pakankamai informacijos, jog internetiniai pirkiniai būtų susieti su blokų grandinėje įvykusiais mokėjimais. Kaip teigiama, tokia informacija gali būti pasidalinta ar net parduodama trečiosiomis šalimis reklamų ar analizės tikslais.

Apibendrinant blokų grandinės skaidrumo ir privatumo principą, galima daryti išvadą, jog dėl decentralizuotumo yra užtikrinamas visiškas transakcijų ir piniginių skaidrumas, tačiau išsaugomo privatumo, dalyvaujant tinkle, atsakomybė atitenka pačiam naudotojui.

Transakcijų nepakeičiamumas. Autorių Gaetani, Aniello, Baldoni, Lombardi, Margheri ir Sassone (2017) aiškinimu, blokų grandinę sudaro iš eilės tarpusavyje susieti blokai su įrašai. Kiekvienas kitas blokas grandinėje turi nuorodą į prieš tai buvusį bloką. Toks tarpusavio blokų susiejimas yra įgyvendinamas kriptografinėmis funkcijomis, o dėl to grandinėje esantys blokai, turintys transakcijų informaciją, yra nepakoreguojamos.

Taigi, apibūdinus blokų sąsają su kitais grandinėje esančiais blokais, galima teigti, jog dėl kriptografinių funkcijų, transakcijos, esančios blokų grandinėje, nebegali būti pakeistos.

1.2. Išmaniųjų sutarčių apžvalga

Blokų grandinė iš pradžių buvo skirta tarpusavio „Bitcoin“ mokėjimams, tačiau įvairesnių panaudojimo atvejų vis daugėja, ne išimtis – išmaniosios sutartys. Luu, Chu, Olickel, Saxena ir Hobor (2016) išmaniąsias sutartis apibrėžia kaip programą, vykdomą blokų grandinėje ir kurios teisingą vykdymą užtikrina konsensuso protokolai. Paprasčiausiuose tokiose programose galima nustatyti įvairias

taisykles, kurias tenkinant, išmanusis kontraktas atliks nustatytus veiksmus, sudėtingesnėse – įgyvendinti įvairias taikomąsias programas, kurios gali apimti didesnius finansinius instrumentus, pavyzdžiui, taupomąsias sąskaitas, bei savarankiško ar autonominio valdymo programas, skirtas, pavyzdžiui, decentralizuotus azartinius lošimus.

Išmanusis kontraktas yra identifikuojamas 160-ies bitų adresu, o vykdomasis kodas – talpinamas blokų grandinėje. Su tokia išmaniąją sutartimi naudotojai sąveikauja siųsdami transakcijas į sutarties adresą. Tai reiškia, kad į blokų grandinės tinklą priimta transakcija, turinti gavėjo – išmanaus kontrakto – adresą, bus vykdoma visų tinklo dalyvių, pasinaudojant įvestimi iš esamos būsenos ir transakcijos informacijos. Išmaniųjų sutarčių išvestis (programos rezultatas) bus nuspręsta tinklo, remiantis konsensuso protokolais. (Luu, Chu, Olickel, Saxena, Hobor, 2021).

Išmaniųjų sutarčių architektūra yra pagrįsta jų nepakeičiamumu. Šį bruožą nagrinėja Salehi, Clark ir Mannan (2022) pastebėdami, kad tai gali būti ir privalumas, kuris užtikrina vientisumą ir skaidrumą, ir klaida, neleidžianti diegti į išmaniuosius kontraktus saugumo atnaujinimų ar naujų funkcijų. Panaudojant programinės įrangos technikas, išmanieji kontraktai gali būti atnaujinti įvairiais metodais. Pirmasis taikomas metodas yra paprasčiausias: įdiegti naują kontraktą į naują adresą, bet tam reikia atnaujinti visas kitas sąsajas, susijusias su atnaujinamu kontraktu. Antrasis būdas – naudoti tarpinį kontraktą (angl. *proxy contract*), kuris priimtų operaciją, bet vykdymą atliktų kitas nustatytas kontraktas, kurį galima pakeisti tarpinėje išmaniojoje sutartyje. Šio metodo minusas yra tas, kad tarpinis kontraktas palaiko tik pradinio kontrakto funkcijas, tad į atnaujinamą kontraktą pridėti funkcijų negalima. Trečiasis variantas apima kontraktų skaidymą į dvi dalis: būsenos ir logikos. Tokių kontraktų susiejimo architektūra yra sudėtingesnė, tačiau suteikia galimybę peradresuoti nežinomas funkcijas iš vieno kontrakto į kitą, kurias atnaujintas kontraktas įgyvendina.

Vienas esminių išmaniųjų sutarčių minusas yra kaina. Šį išmaniųjų sutarčių iššūkį apžvelgia Kostamis, Sendros ir Efraimidis (2021) teigdami, kad saugoti duomenis išmaniuosiuose kontraktuose Ethereum blokų grandinėje yra brangu, tačiau pridėdami, kad decentralizuotos aplikacijos naudoja įvairius metodus, siekiant sumažinti kainą. Siekiant optimizuoti kaštus, galima taikyti hibridinį modelį, apimantį *on-chain* ir *off-chain* architektūrą. Tokiu atveju duomenys yra saugomi decentralizuotose saugyklose, pavyzdžiui, IPFS ir Swarm, o išmaniuosiuose kontraktuose naudojamos nuorodos į duomenis. Norint saugoti duomenis blokų grandinėje, taikomi pigūs metodai: pirma, duomenis galima saugoti įvykių žurnaluose (angl. *event-logs*), antra, galima naudoti transakcijų apkrovą (angl. *transaction payload*), trečia, panaudoti nenaudojamų funkcijų parametrus (angl. *unused function arguments*).

Naudojantis šiais metodais, informacija yra saugoma blokų grandinėje, tačiau neapkrauna kontrakto, o dėl to mažėja kaštai.

Išmaniųjų sutarčių realizavimas pasižymi technologiniais iššūkiais. Taherdoost (2023) apžvelgia tokių kontraktų realizacijos sunkumais: pirmiausia, kadangi išmaniosios sutartys negali būti pakeistos dėl saugumo, išmaniųjų sutarčių kūrėjai privalo užtikrinti kokybišką, patikimą programinį kodą prieš pateikiant sutartį į blokų grandinę. Atnaujinti jau esamą sutartį, siekiant patobulinti ar ištaisyti trūkumus, gali būti taip pat sudėtinga. Antra, išmaniosios sutartys, vykdydamos kodą, naudoja skaičiavimo resursus, o tai reiškia, kad kiekviena operacija sutartyje kainuoja. Dėl šios priežasties, siekiant mažinti sutarčių operacijų vykdymo kainą, reikalinga taip pat ir optimaliai sukurti išmaniąją sutartį aprašantį kodą.

Vertinant išmaniųjų sutarčių kūrimo iššūkius, Parizi, Singh ir Dehghantanha (2018) analizuoja naudojamų programavimo kalbų paskirtį, prieinamumą ir patogumą. Pažymima tai, kad kiekviena aptariama programavimo kalba – Solidity, Pact ir Liquidity – atsirado, siekiant išspręsti specifines dalykinės srities problemas. Pavyzdžiui, Solidity pasižymi savo paprastumu naujiems išmaniųjų sutarčių kūrėjams, tačiau Pact ir Liquidity, nors ir yra sudėtingesnės realizacijos prasme ir turi mažesnį pritaikomumą, jos užtikrina geresnį saugumą, sumažina galimybes kūrėjams palikti klaidų ar pažeidžiamumų. Minėtų programavimo kalbų, naudojamų aprašant išmaniąsias sutartis, palyginimas pateikiamas 1 lentelėje.

1 lentelė

Išmaniųjų sutarčių programavimo kalbų palyginimas

Programavimo kalba (Platforma)	**Pagrindiniai bruožai**	**Privalumai**	**Trūkumai**
Solidity (Ethereum)	Orientuota į prieinamumą kūrėjams	Paprasta sintaksė, palaikomas paveldėjimas, bibliotekos	Didelė tikimybė palikti klaidų ar pažeidžiamumų
Pact (Kadena)	Specifinės srities problemų sprendimas	Į saugumą orientuota dizainas, atominis kodo vykdymas (transakcijos)	Pasižymi sudėtinga realizacija
Liquidity (Tezos)	Specifinės srities problemų sprendimas	Aukšto lygio tipai, lokalūs kintamieji	Mažas pritaikomumas rinkoje

Šaltinis: sudaryta autoriaus, remiantis Parizi, R. M., Singh, Amritraj, Dehghantanha, Ali (2018). Smart Contract Programming Languages on Blockchains: An Empirical Evaluation of Usability and Security.

Nors išmaniosios sutartys pasižymi savo saugumu, nepakeičiamumu, egzistuoja architektūriniai sprendimai, leidžiantys manipuliuoti kontraktais. Jų kūrimas kelia specifinių iššūkių, pavyzdžiui, efektyvumo (vykdymo kainos) ir saugumo (programinio kodo atžvilgiu), tačiau jos gali būti išspręstos pasirenkant tinkamus metodus.

Išmaniųjų sutarčių plitimas ne tik parodo, kad tai yra pažangi technologija, tačiau atskleidžia ir pažeidžiamumus. Kaip pastebi Luu, Chu, Olickel, Saxena ir Hobor (2021), kadangi išmanieji kontraktai veikia viešuose tinkluose, šios atsiduria priešiškų rinkos dalyvių akiratyje, siekiant pažeisti ir manipuliuoti išmaniosiomis sutartimis. Nevertinant prieš tai aptartų technologinių iššūkių, su kuriais susiduria kūrėjai, aprašantys išmaniąsias sutartis, egzistuoja ir kitos pažeidžiamumo atakos: priklausomybės nuo transakcijų eilės ar laiko žymių, netinkamai apdorotų kodo išimčių, pakartotinio vykdymo.

Išmaniųjų kontraktų vykdymas ir rezultatai gali būti priklausomi nuo aplinkos, kurioje jie yra vykdomi. Pirmiausia, kontraktai, priklausantys nuo transakcijų eilės (angl. *transaction-ordering dependence*), gali sukelti netikėtą vykdymo rezultatą, jei dvi ar kelios transakcijos, vykdomos išmaniajame kontrakte, bus sukeistos vietomis. Dažniausiai tai gali būti netyčinis blokų grandinės

veiksmas, kuomet, pavyzdžiui, vienas rinkos dalyvis atnaujina kontraktą, o kitas eilės tvarka bando gauti informaciją iš išmaniojo kontrakto, tad sukeitus transakcijas vietomis, išmaniojo kontrakto informacija bus netiksli, nes kontraktas jau turės pakeistą būseną. Tai reiškia, kad vartotojai gaus kitokį atsaką, nei šie tikėjosi. Ši ataka gali būti išnaudota ir neteisėtiems veiksmams, kai manipuliuojama transakcijų eiliškumu, ir gali padaryti žalą, siekiant uždirbti pelną ar prieiti prie lėšų. Antra, kuomet kontraktas priklauso nuo laiko žymos (angl. *timestamp dependency*), kasėjai gali manipuliuoti, nors ir nedideliame rėžyje, transakcijos laiku. Tai gali sukelti netinkamą išmaniojo kontrakto sąlygų vykdymą ir netikslius rezultatus. (Luu, Chu, Olickel, Saxena, Hobor, 2021).

Trečiasis galimas pažeidžiamumas išmaniuosiuose kontraktuose yra susijęs su netinkamu klaidų apdorojimu kontraktuose (angl. *mishandled exceptions*). Kontraktams sąveikaujant tarpusavyje, jų vykdymas gali nutrūkti dėl įvairių priežasčių, pavyzdžiui, esant nepakankamam apmokėjimui, viršijant palaikomam kvietimų medžio dydžiui, kuris Ethereum virtualioje mašinoje (trumpina EVM) yra apribotas iki 1024-ių. Tai sudaro sąlygas manipuliuoti kontraktų rezultatais. Pavyzdinė ataka galėtų apimti 1023 menkiavertes užklausas į kontraktą, o paskutinė – reali užklausa veiksmui atlikti. Kadangi yra pasiekiamas Ethereum virtualios mašinos limitas, dėl netinkamai paruoštų išmaniųjų kontraktų būsena tik dalyje susijusių kontraktų gali būti teisingai pakeičiama, o dalis - neįvykdoma. Tai sudaro sąlygas, pavyzdžiui, perimti svetimą kontraktą ir jį išnaudoti dėl savos naudos (Luu, Chu, Olickel, Saxena, Hobor, 2021).

Ketvirtoji ataka – pakartotinio įvykdymo (angl. *reentrancy*) – išmaniuose kontraktuose yra gana žinoma. Ataka paremta principu, kuomet kontraktas kviečia kitą kontraktą ir laukia įvykdymo atsakymo iš pastarojo. Vykdymo momentu kviečiantysis kontraktas nebus pakeitęs savos būsenos, o senąją būseną galima išnaudoti pakartotine transakcija. Aiškinant pavyzdine atakos galimybe, jei kontraktas susijęs su balansų tikrinimu ir pinigų siuntimu, galima atlikti dvi pinigų siuntimo operacijas už tą patį balansą, nes pirmojo vykdymo metu balansas nebus atnaujintas (sumažintas) antrajam vykdymui. (Luu, Chu, Olickel, Saxena, Hobor, 2021).

Išmaniosios sutartys, priklausomai nuo realizacijos būdo ir priklausomybių nuo aplinkos, gali būti pažeidžiamos manipuliuojant kontraktų būsena ir operacijų vykdymo laiku, o dėl to gali atlikti nekorektiškas operacijas.

Išmaniosios audito procedūros apima autonominius vidaus kontrolės testus ir analitines procedūras, kurios yra diegiamos blokų grandinėje. Rozario ir Vasarhelyi (2018) įžvalgomis, tai leidžia beveik realiuoju laiku teikti finansines ataskaitas kitoms suinteresuotoms šalims, pavyzdžiui, investuotojams, tiekėjams, inspektoriams, vertybinių popierių biržoms. Kadangi suteikiama galimybė

realiu laiku taikyti išmaniąsias audito procedūras, šios turi didelį potencialą aukštesnei audito kokybei pasiekti, suteikia efektyvumo auditorių darbui. Dėl to, kad ataskaitos yra platinamos viešai, jos padeda patenkinti augančius poreikius didesniam skaidrumui ir savalaikiam ataskaitų teikimui užtikrinti.

Nagrinėjant sritis, kur išmaniosios audito procedūros galėtų pasireikšti savo privalumais, išskiriama gamybos pramonė, jų finansinių srautų monitoringas. Šioje srityje išmanieji kontraktai apibrėžia įmonių ir vykdymo įsipareigojimus – sąlygos, kaip verslo logika, yra užprogramuojamos ir diegiamos blokų grandinėje. Platformoje veikiančios išmaniosios sutartys suteikia interesuotoms šalims vertinti vykdomus sandorius, jų būseną, o esant reikalui – vykdyti įsipareigojimų patikrinimą. Tikrinimo metu galima išryškinti bėdas, dėl kurių sandoriai vykdomi nesąžiningai, nesilaikant taisyklių. Pažeidimus vertinant autonomiškai, sandoriai gali būti apribojami. Tokiu būdu išmaniosios audito procedūros ne tik padidina skaidrumą, tačiau ir sudaro galimybes efektyvesniam, patikimesniam rizikų valdymui bei leidžia užkirsti kelią nesąžiningam sandorių vykdymui. (Rozario, Vasarhelyi, 2018).

Nors yra pabrėžiamos limituotos auditavimo praktinės taikymo galimybės, išmaniųjų kontraktų technologija, kaip priemonė, skirta išmaniosioms audito procedūroms vykdyti, plėtojama eksponentiškai. Išoriniam auditavimui yra logiška sekti technologinius pasikeitimus ir tobulėjimą srityje, kad būtų išlaikytas pažangumas moderniame ir skaitmeniniame pasaulyje, tad svarbu vertinti kaip transformuojasi tradicinės išorės audito modelis, atsižvelgiant į technologinius pokyčius. (Rozario, Vasarhelyi, 2018).

Išmaniosios audito procedūros gali būti įgyvendintos su išmaniaisiais kontraktais, tačiau pabrėžiama taikymo sritis yra verslo logikai užtikrinti, sudaromų sandorių blokų grandinėje auditavimo galimybės neaptariamos.

1.3. Automatizuotos prekybos strategijos ir architektūra

Technologijų vystymasis iš esmės pakeitė tai, kaip vykdoma prekyba biržose. Pasak Jones (2013), prieš kompiuterių atsiradimą vertybinių popierių ar kitų finansinių priemonių prekyba būdavo vykdoma žmonių, gyvai esančių prekybos salėje, tačiau prekybos procesas pasikeitė dėl automatizacijos galimybių.

Jungtinių Amerikos Valstijų vertybinių popierių ir biržos komisija automatizuotus prekybininkus charakterizuoja penkiais požymiais. Visų pirma, tokie rinkos dalyviai naudoja sudėtingas ir itin sparčias kompiuterines programas tam, jog būtų efektyviai sugeneruojami, nukreipiami ir vykdomi prekybiniai veiksmai. Siekiant aukšto našumo, vykdant tokias operacijas, svarbus ir antrasis požymis – naudojimasis kolokacijos paslaugomis ir individualiais biržų teikiamais duomenų kanalais, kuriais siekiama sumažinti, pavyzdžiui, tinklo, delsos laiką. Trečiuoju charakterizavimu yra išskiriami pozicijų

sudarymo ir likvidavimo terminai – šie būna labai trumpi. Automatizuotoje prekyboje taip pat svarbus prekybos dienos užbaigimas su minimaliai neuždarytų pozicijų, kad šios nebūtų laikomos, pavyzdžiui, per naktį, bei daugybės finansinių operacijų pateikimas ir uždarymas – atitinkamai ketvirtasis ir penktasis bruožai. Bendruoju atveju, tokias savybes turinčius prekybininkus galima įvardinti kaip profesionalus, kurie naudodami įvairias strategijas kasdien sudaro daug finansinių sandorių (Jones, 2013).

Pasak Gomber, Arndt, Lutat ir Uhle (2011), prekyba vertybiniais popieriais elektroninėje erdvėje prasidėjo praėjusio amžiaus aštuntame dešimtmetyje, kuomet finansinių įmonių reguliavimo institucija Jungtinėse Amerikos Valstijose pradėjo naudoti kompiuterija grįstą automatinio kotiravimo sistemą, kuri šiandien vadinama NASDAQ. Europoje kompiuterizuotos biržos pradėjo teikti paslaugas devintajame dešimtmetyje, tačiau tik po dešimtmečio prekyba vertybiniais popieriais buvo vykdoma visiškai automatizuotose elektroninėse biržose. Tai lėmė, jog prekyba fizinėse biržose itin sumažėjo, daug prekybos proceso etapų buvo automatizuoti. Dėl atsiradusių galimybių ir pasikeitusios situacijos, didelę dalį rinkoje pradėjo užimti algoritminė bei automatizuota prekyba.

Automatizuotoje prekyboje nėra vienos bendros prekybos strategijos, kurią naudotų visi ar dauguma algoritminių prekybininkų. Automatizuotai prekybai veikimo esmei paaiškinti galima naudoti šias strategijas: veikimas kaip kotiruotoju (rinkos formavimas), aukšto dažnio santykinės vertės prekyba bei kryptinga prekyba, paremta naujienų pranešimais, operacijų srauto ar kitais signalais (Jones, 2013).

Rinkos formavimas. Kotiruotojai tuo pačiu metu pateikia finansines operacijas pasiūlymui pirkti (angl. *bid*) ir parduoti (angl. *ask*). Tokiu būdu yra veikiama kaip tarpininkais, kurie suteikia likdivumą rinkoje. Tai leidžia kitiems rinkos dalyviams greitai pirkti arba parduoti vertybinius popierius ar akcijas. Veikiant rinkos formuotoju, yra siekiama uždirbti skirtumą tarp pasiūlymo pirkti ir pasiūlymo parduoti kainų – dažnai vadinamu *bid ask spread*. Rinkos dalyviai – kotiruotojai – dažnai atnaujina savo kotiruotes kaip reakciją į kitų rinkos dalyvių vykdomas operacijas, kainų judėjimus. Šis nuolatinis atnaujinimo procesas padeda išlaikyti konkurencingas kainas rinkoje bei sumažinti riziką prarasti pinigus geriau informuotiems rinkos dalyviams. Tai reiškia, kad viena iš svarbiausių šios strategijos dalių yra informacija ir su ja susijusi rizika. Geriau informuoti rinkos dalyviai dažniausiai turi geresnę informaciją apie būsimas kainų tendencijas, todėl kotiruotojams svarbu kuo daugiau aktualios informacijos, susijusios su prekiauja finansine priemone, įtraukti į savo finansines operacijas, kad būtų sumažinta rizika atidarytai pozicijai, jei rinkos judėjimas būtų priešingas. Dėl šios priežasties, šiai strategijai yra svarbus didelis kotiruočių atnaujinimo dažnis. Algoritminiai prekybininkai gali pateikti ir atšaukti daug operacijų, kol įvyksta sandoris (Jones, 2013).

Apibendrinant rinkos formavimo strategiją, naudojant didelio našumo algoritmus siekiama uždirbti pelną tarp pasiūlymo pirkti ir pasiūlymo parduoti kainos skirtumų.

Kryptinga prekyba. Ši strategija remiasi sparčia analize, siekiant nustatyti įvairius signalus, kuriuos galima panaudoti vykdant prekybą. Signalų paieška gali būti pritaikoma sekant operacijų srautus, bandant nustatyti kitų rinkos dalyvių elgesį rinkoje (Jones, 2013).

Finansinių operacijų srauto stebėjimas gali padėti nustatyti būsimą kainos pokytį. Jei kiti rinkos dalyviai vykdo dideles operacijas už esamas siūlomas kainas, ši strategija gali daryti išvadą, kad kiti rinkos dalyviai turi palankios informacijos. Automatizuotoje prekyboje naudojami algoritmai siekiant išsiaiškinti tokius dėsningumus gali pritaikyti automatinę reakciją ir pirkti tokias akcijas, tikėdamasis kainos kilimo. Kita šios strategijos variacija – stebėjimas didelių operacijų per trumpą laiką, siekiant toliau didinti vertybinių popierių kainą, realizuojant pelną pardavus akcijas su pakilusia kaina. Šios strategijos variacijos – operacijų dydžio ir kiekio sekimas – gali būti apsunkintas analizės prasme, kadangi kiti rinkos dalyviai gali skaidyti dideles operacijas į mažesnes, naudotis „tamsiaisiais fondais“ vykdomų operacijų paslėpimui tam, kad žinoma informacija apie įmonių ketinimus būtų paslėpta nuo automatizuotų prekybininkų (Jones, 2013).

Apibendrinant kryptingos prekybos strategiją, siekiama analizuojant vykdomas operacijas nuspėti galima kainos tendenciją rinkoje ir trumpalaikėje perspektyvoje uždirbti pelną iš būsimų kainos pokyčių.

Apibendrinant algoritminės prekybos strategijas, siekiama nustatyti finansinių instrumentų kainų neatitikimus, numatyti būsimus pokyčius rinkoje, siekiant uždirbti pelną iš kainų skirtumų netolimoje ateityje.

1.4. Nesąžiningos prekybos rizikos ir tipiniai pažeidimai blokų grandinėje

Nors blokų grandinės principai užtikrina saugumą, tačiau egzistuoja galimybės manipuliuoti rinka. Kaip pastebi S. Yang, Zhang, Huang, Chen, Y. Yang ir Zhu (2022), įtakingi rinkos dalyviai, tokie kaip „kasėjai“ ar galingos automatizuotos sistemos (toliau – užpuolikais), gali ženkliai padidinti savo uždirbamus pelnus, kuomet, tinkamai pasirinkus vykdomas kitų rinkos dalyvių – potencialių aukų – transakcijas, prieš šias įterpia, išmeta ar pertvarko jų eilę blokuose. Toks pelno gavimo metodas vadinamas maksimaliai išgaunama verte (angl. *maximal extractable value*), kuris turi gana didelę reikšmę blokų grandinės saugumui ir decentralizacijai. Šią strategiją galima skirstyti į tris kategorijas: transakcijų įterpimu prieš aukos transakciją (angl. *frontrunning*), po aukos transakcijos (angl. *backrunning*) ir kelių transakcijų įterpimą prieš ir po aukos transakciją (angl. *sandwich*).

Transakcijos įterpimo prieš nukentėjusio rinkos dalyvio transakciją (angl. *frontrunning*) ataką galima skaidyti į dvi formas. Pirmoji forma susideda iš principo, kad užpuolikas moka itin didelį transakcijos mokestį tam, kad jo transakcija būtų atlikta greičiau nei kitų rinkos dalyvių. Pavyzdžiui, pastebėjus už klaidingai žemą kainą parduodamą skaitmeninį turtą ir sumokėjus didelį mokestį „kasėjams“, padidinami šansai, kad užpuoliko vykdoma pirkimo transakcija bus vykdoma greičiausiai, taip aplenkiant kitus rinkos dalyvius. Kita forma – transakcijos įterpimas prieš aukos transakciją ne tam, kad būtų laimėtas sandorio įsigijimo pirmenybė, kaip įvardinta ką tik minėtoje formoje, bet kad būtų paruošiama aukos transakcija kitai artėjančiai atakai (S. Yang, Zhang, Huang, Chen, Y. Yang, Zhu, 2022).

Ataka, kai užpuoliko transakcija įterpiama po aukos transakcijos, siekiama pasinaudoti arbitražo galimybe. Aptikus aukos transakciją vienoje biržoje, kuri tikėtina stipriai išaugins skaitmeninio turto kainą, užpuolikas perka turtą iš kitos biržos, kurią parduoda pirmojoje biržoje iškart po aukos transakcijos, taip uždirbant iš kainų skirtumo. Nors ši ataka nedaro tiesioginės žalos aukai, ja yra pasinaudojama (S. Yang, Zhang, Huang, Chen, Y. Yang, Zhu, 2022).

Atakuojant rinkos dalyvių transakcijas iš abiejų pusių, t. y. kai užpuolikas įterpia transakciją prieš ir po aukos transakcijos, vadinamoje „sumuštinio“ atakoje (angl. *sandwich*) siekiama manipuliuoti rinka taip, kad būtų išnaudota didėjanti kaina. Užpuolikas pirkdamas tą patį turtą prieš auką ir iškart parduodamas po aukos sandorio įvykdymo, uždirba iš aukos išprovokuotos kainos kilimo, o kadangi užpuolikas pirko tą patį turtą iškart prieš aukos sandorį, auka patiria nuostolį iš užpuoliko sukelto kainos šuolio (S. Yang, Zhang, Huang, Chen, Y. Yang, Zhu, 2022).

Remiantis šiomis maksimaliai išgaunamos vertės strategijomis automatizuotoje prekyboje, susiduriama su saugumo praradimu. Šios atakos ne tik pasinaudoja kitais rinkos dalyviais, kuomet užpuolikai tiesiogiai uždirba iš aukų ir daro joms finansinius nuostolius, bet paveikia visą blokų grandinę. Pirmiausia, pažangios automatizuotos prekybos sistemos konkuruoja tarpusavyje, siekiant naudos. Tai apkrauna visą blokų grandinės tinklą bei didina transakcijų mokesčius visiems rinkos dalyviams. Antra, užpuolikai siūlydami didesnį mokestį už prioritetines transakcijas, skatina „kasėjus“ nutolti nuo sąžiningos prekybos ir rinktis jiems palankias transakcijas, didinančias uždarbį. Tai tampa įmanoma manipuliuojant „dujų“ (angl. *gas*) – resursų matavimo vieneto, reikalingo skaičiavimams atlikti – kaina. Trečia, dėl didelio vykdymo masto, siekiant iš strategijos išgauti maksimalią naudą, didėja blokų grandinės centralizuotumas, o kartu mažėja ir rinkos skaidrumas. (S. Yang, Zhang, Huang, Chen, Y. Yang, Zhu, 2022).

Maksimaliai išgaunamos vertės metodai leidžia manipuliuoti rinka, pertvarkant transakcijų eiliškumą. Dėl šių atakų nukenčia tiek tiesioginiai rinkos dalyviai, kurių sąskaita yra uždirbama, tiek visi rinkos dalyviai, kuriems didėja mokesčiai bei mažėja skaidrumas.

Egzistuojantys metodai, skirti atpažinti maksimaliai išgaunamos vertės atakoms, pasak Park, Jeong, Y. Lee, Son, Jang, J. Lee (2023), susiduria su problema, kad šie yra priklausomi nuo trečiųjų šalių įrankių, skirtų sekti sandorius blokų grandinėje, yra per daug konservatyvūs algoritmai, kurie nesugeba identifikuoti atakų, bei yra prastai palaikomi rinkoje atsirandant vis daugiau decentralizuotų finansų. Grafinių neuroninių tinklų pagrindu sukurtas įrankis, skirtas identifikuoti atakas, pasižymi dideliu tikslumu ir reikalauja mažų vykdymo resursų. Pristatytas modelis taip pat išsprendžia egzistuojančių sprendimų problemas: priklausomybe nuo trečiųjų šalių, pritaikomumo ir palaikymo skirtingoms ir vis atsirandančioms blokų grandinėms.

Grafinių neuroninių tinklų grįstas modelis gali aptikti atakas efektyviau nei tradiciniai algoritmai, kurie yra priklausomi nuo trečiųjų šalių ir sunkiai palaikomi.

Siekiant pažaboti maksimaliai išgaunamos vertės strategiją, decentralizuotos biržos gali pradėti taikyti grupinių aukcionų (angl. *batch auctions*) metodą. Pasak Zhang, Li, Sun, E. Chen ir X. Chen (2025), šis būdas leistų sandorių transakcijas apdoroti ne individualiu būdu, o trumpalaikėje perspektyvoje sujungti jas visas į grupę pagal bendrą kainą ir sandorio kryptį. Tai reiškia, kad visos sugrupuotos transakcijos yra atliekamos už tą patį kursą, o dėl to sumažėja galimybės taikyti maksimaliai išgaunamos vertės atakas, taip išvengiant transakcijų eilės pertvarkymo.

Apibendrinant, grupinių aukcionų metodas apdoroja daugybę transakcijų už vienodą kainą, kad būtų sumažinama maksimaliai išgaunamos vertės strategijos atakos tikimybė.

2. APSAUGOS NUO AUTOMATIZUOTOS PREKYBOS ATAKŲ SISTEMOS TECHNINIS PASIŪLYMAS

Šiame skyriuje pristatomas pagrįstas apsaugos nuo automatizuotos prekybos atakų sistemos techninis pasiūlymas.

2.1. Skaidrumo ir piktnaudžiavimo poreikis

Reguliacinis reikalavimas – rinkos skaidrumo ir piktnaudžiavimo prevencijos užtikrinimas – yra įtvirtintas Europos Parlamento ir Tarybos reglamente dėl kriptoturto rinkų (angl. *Markets in Crypto-Assets, MiCA*) (2023). Reglamento devyniasdešimt pirmajame straipsnyje, pavadinimu „Draudimas manipuliuoti rinka“, uždraudžia tiek bandyti, tiek vykdyti rinkos manipuliaciją. Toks pažeidimas, remiantis reglamentu, apima bet kurį iš veiksmų: suteikiami netinkami signalai dėl paklausos ir kainos, ar ši fiksuojama kaip nenormali arba dirbtinė, skleidžiama melaginga ar klaidinanti informacija arba vykdomi veiksmai, darantys poveikį vieno ar kelių kriptoturtų vienetų kainai. Tokie veiksmai iškreipia rinką, trikdo realius sandorius, todėl finansų sektoriaus subjektams kyla pareiga pagal to paties reglamento devyniasdešimt antrojo straipsnio „Piktnaudžiavimo rinka prevencija ir nustatymas“ įgyvendinti rinkos stebėjimo priemones, kurios bandytų užkirsti kelią piktnaudžiavimui rinka ir jį nustatytų. Be to, reglamentas įpareigoja tokius subjektus teikti informaciją apie pagrįstus įtarimus atitinkamoms institucijoms. Tai reiškia, kad reglamentas ne tik griežtai draudžia rinkos manipuliaciją, bet ir įpareigoja rinkos dalyvius rinkti bei teikti informaciją apie tokią veiklą reguliuojančioms institucijoms.

Nors MiCA reglamentas sukuria teisinį pagrindą rinkos skaidrumui, praktinis šių nuostatų įgyvendinimas reikalauja specifinių priemonių. Siekiant įgyvendinti reglamento 92-ąjį straipsnį, kad būtų ne tik nustatomos rizikos, bet ir užkertamas kelias piktnaudžiavimui, būtina diegti sprendimus, kurie būtų integruoti į pačią prekybos infrastruktūrą. Tai reiškia, kad atitiktis tampa neatsiejama dalimi nuo saugumo architektūros, kuri gali techniškai blokuoti rinkos manipuliacijas dar prieš joms įvykstant.

2.2. Išmaniųjų kontraktų tinkamumas

Esminis prevencinės sistemos iššūkis yra užtikrinti saugų ir skaidrų lėšų valdymą decentralizuotoje aplinkoje. Tradiciniuose (centralizuotuose) sprendimuose vartotojui tektų itin pasitikėti sistema: perduoti lėšas neaiškiems procesams, o tai sukuria saugumo spragą ir neatitinka iškeltų saugumo reikalavimų. Dėl šios priežasties reikalingas decentralizuotas sprendimas – išmaniosios sutartys, galinčios veikti kaip saugus tarpininkas tarp decentralizuotų biržų ir vartotojų.

Pirmasis svarbus išmaniųjų sutarčių pasirinkimo aspektas yra jų nekintamumas ir patikimumas. Kaip pabrėžia Alharby ir Moorsel (2017), kai tik kontraktas yra sukuriamas ir įdiegiamas į blokų grandinę, išmaniosios sutarties kodas nebegali būti pakeistas. Dėl šios priežasties išmaniosios sutartys tampa svarbiu pasirinkimu dėl determinizmo ir skaidraus tarpininkavimo. Tai reiškia, kad vartotojai žino tiksliai, ką išmanioji sutartis daro, nes jos kodas negali būti pakeistas ir tapti kenkėjiškas, bei vartotojai gali būti ramūs dėl to, kad lėšos bus panaudotos tikslingai – apsaugotam sandorių vykdymui decentralizuotose biržose ar, atakos atveju, grąžinimui.

Procesų nedalomumas ir automatizavimas taip pat yra svarbus pasirinkimo aspektas. Išmaniųjų sutarčių paskirtis yra ne tik išlaikyti viešą ir nekintamą kodą, bet ir užtikrinti, kaip pastebi Christidis bei Devetsikiotis (2016), automatizuotą kelių žingsnių procesų vykdymą. Tai reiškia, kad išmanieji kontraktai leidžia sujungti skirtingus procesus į vieną nedalomą vykdymo grandinę. Vartotojams tai aktualu dėl to, kad sudėtingi procesai, apimantys lėšų perėmimą, sąveiką su decentralizuota birža ir galutinį atsiskaitymą, bus įgyvendinti be incidentų, taip padidinant pasitikėjimą sistema.

Išmanieji kontraktai turi patikimą tarpininkavimo architektūrą. Werner, Perez, Gudgeon, Klages-Mundt, Harz ir Knottenbelt (2021) šį privalumą aiškina dalyvių galimybe išlaikyti pilną savo lėšų kontrolę bet kuriuo laiko momentu. Tai reiškia, kad sistema iš esmės niekada neturi tiesioginės prieigos prie vartotojo lėšų bei jų laisvo disponavimo. Toks sprendimas leidžia pašalinti trečiosios šalies riziką: jei sistema yra pažeidžiama, pavyzdžiui, jos kiti moduliai, esantys ne grandinėje, vartotojų lėšos lieka saugios, jei išmaniajame kontrakte nenumatytos techninės galimybės atlikti pervedimus į pašalinius adresus.

Apibendrinant išmaniųjų kontraktų tinkamumą, šie yra esminė sąlyga, siekiant užtikrinti nenutrūkstamą pasitikėjimą sistema. Išmaniųjų sutarčių pasirinkimas suteikia nekintamumo, procesų nedalomumo bei skaidraus tarpininkavimo privalumus decentralizuotoje aplinkoje.

2.3. Priešakinės transakcijos atakos svarbumas

Automatizuotos priešakinės atakos decentralizuotose finansų sistemose daro nemažą įtaką ir bendrai rinkai, ir kiekvieno atskiro rinkos dalyvio patirčiai. Tai reiškia, kad prekyba su galimybe perrikiuoti transakcijas blokuose turi įtaką transakcijų rezultatams.

Kitaip nei kitos didelio dažnio prekybos, priešakinė ataka yra laikoma nesąžininga. Anot Eskandari, Moosavi ir Clark (2019), taip yra dėl to, kad kitos didelio dažnio prekybos atakos, pavyzdžiui, arbitražas, yra legalus ir skatinamas, kadangi tai padeda rinkai greičiau apsikeisti naujausia informacija apie kainas, tuo tarpu priešakinė ataka yra veiksmas, kuomet užpuolikas pasinaudoja auka (ją apvogia),

siekiant uždirbti, o tai daro žalą konkretiems rinkos dalyviams, bet nepadeda pačiai rinkai. Dėl šios priežasties priešakinės atakos užkardymui turi būti skiriamas dėmesys, siekiant jos žalos minimizavimo.

Priešakinės transakcijos ataka yra svarbi ir tuo, kad yra sudedamoji kitų atakų dalis. Zhou, Qin, Torres, Le bei Gervais (2020) pabrėžia tai, kad tai yra fundamentalus komponentas, sudarantis pagrindą rinkos manipuliacijoms, pavyzdžiui, dvipusio įterpimo atakai. Ši algoritminė ataka po aukos transakcijos turi papildomą transakciją, kuria realizuojamas pelnas, pasinaudojant aukos sukeltu kainos pokyčiu. Tai reiškia, kad jeigu prevenciškai galima identifikuoti ir apsisaugoti nuo priešakinės transakcijos atakos, galima apsisaugoti ir nuo kitų atakų. Dėl šios priežasties priešakinės transakcijos prevencija yra kritinė grandis, kuri suteikia plačiausią apsaugos padengimą.

Svarbus algoritminės prekybos prevencijos bruožas yra tas, kad ataką galima pastebėti ir sustabdyti realiu laiku. Skirtingai nei kitos didelio dažnio atakos, tokios kaip arbitražas, kuris, anot Qin, Zhou ir Gervais (2021), remiasi skirtingų rinkų stebėjimu, tad reikia stebėti visą tinklo būseną, norint numanyti bandymą pasinaudoti arbitražu. Tuo tarpu priešakinių transakcijų ataka yra pagrįsta bandymu pasinaudoti viena konkrečia transakcija, kuomet ši yra dar nepatvirtinta ir neįtraukta į bloką. Tai reiškia, kad jei užpuolikas viešai stebi ir analizuoja galimus bandymus atakuoti ir juos vykdo, prevencinės sistemos, veikiančios tame pačiame lygyje, gali reaguoti į užpuolikų veiksmus lygiai tokiomis pačiomis sąlygomis. Tad bandymą manipuliuoti rinka galima bandyti sustabdyti dar prieš įtraukiant transakcijas į bloką.

Žala ir nuostoliai, kuriuos sukelia priešakinė transakcijos ataka yra pakankamai didelė. Remiantis analizuotais adresais ir išmaniaisiais kontraktais, Qin, Zhou, Gervais (2021) pavyko nustatyti apie 750 tūkstančių sėkmingų dvipusio įterpimo tipo atakų, kurių sudedamoji dalis yra ir priešakinė transakcijos ataka. Įvairiose decentralizuotose biržose padaryta žala vertinama šiek tiek daugiau nei 174 milijonai JAV dolerių per 32 mėnesius. Tai reiškia, kad vidutinė vienos atakos žala (arba užpuoliko uždirbamas pelnas) siekia apie 230 JAV dolerių. Dėl tokio didelio masto žalos ir aukšto vidutinio pelningumo, atakas vykdyti yra patrauklu, tad reikalingi sprendimai, galintys sumažinti padarinius.

Apibendrinant algoritminės priešakinės transakcijos atakos ypatumus, ši išsiskiria dideliais finansiniais nuostoliais rinkos dalyviams bei sąžiningos prekybos principų pažeidimu, todėl jos prevencija yra itin svarbi užduotis, siekiant decentralizuotų finansų sistemos saugumo.

2.4. Aukos apsaugojimo galimybės

„Ethereum“ tinkle dominuoja transakcijų rikiavimo logika, kuri prioretizuoja transakcijas pagal dujų kainą. Pasak Torres ir Camino (2021), „kasėjai“ sprendžia, kurias transakcijas pasirinkti ir kokia eile įtraukti jas į bloką, tad „kasėjai“ bando pasididinti savo užmokestį pasirinkdami transakcijas

pagal vykdymo kainą – tai liečia net 95 procentus rinkoje esančių „kasėjų“. Mokesčio dydžio prioretizavimu grindžiamas ir „replace-by-fee“ mechanizmas. Narayanan, Bonneau, Felten, Miller ir Goldfeder (2016) nurodo, kad, jeigu tinkle atsiranda konfliktuojančios transakcijos, „kasėjai“ pasirinks tą, kuri moka didesnį mokestį. Tai grindžiama tuo, kad trumpalaikėje perspektyvoje „kasėjui“ atlikti tokį veiksmą yra ekonomiškai naudinga. Tai reiškia, jog esant kelioms transakcijoms, kurią atlieka tas pats adresas su tuo pačiu eilės numeriu, bus vykdoma transakcija su didesniu mokesčiu. Kadangi tinkle jau esančių nepatvirtintų transakcijų atšaukti negalima, pritaikant „replace-by-fee“ mechanizmą galima pakeisti transakciją, kuri atliktų kitą veiksmą, pavyzdžiui, atliktų kitą išmanaus kontrakto metodo kvietimą.

Nepatvirtintų transakcijų pakeitimas tinkle turi didelį pranašumą siekiant apsaugoti automatizuotos prekybos strategijų aukas – vykdymo laiką. Eskandari, Moosavi ir Clark (2019) aptaria kitus apsaugos būdus: „commit/reveal“ ir „call market“. Pirmasis apsaugos būdas yra dviejų procesų protokolas, o tai reikalauja privalomos laiko delsos tarp transakcijos įsipareigojimo ir atskleidimo. Dėl reikalingo laiko tarpo rinkos dalyviai negali vykdyti pasirinktų transakcijų norimu metu. Antrajame „call market“ metode transakcijos yra surenkamos laike, grupuojamos ir vykdomos kartu, tad iki bendro transakcijų vykdymo susidaro dirbtinė delsa. Taigi, šie abu apsaugos būdai sukelia priverstinę vykdymo delsą, kuri gali lemti rinkos situacijos pasikeitimą, o tai gali pažeisti rinkos dalyvių interesus.

Kitas svarbus bruožas – nepriklausomybė nuo centralizuotų tarpininkų. Standartiškiausia apsauga nuo automatizuotų prekybos strategijų atakų yra privatūs transakcijų kanalai. Kaip aiškina Capponi, Jia ir Wang (2022), „Flashbots“ ir „Eden Network“ sistemos pristatytos su tikslu suteikti objektyvią apsaugą nuo „frontrunning“ atakų ir sumažinti rinkos dalyviams neigiamą poveikį, kuris yra sukeliamas didelių transakcijų kaštų, kuriuos lemia arbitražininkai. Analizuojant apsaugos sistemas, nustatyta, kad ne visi „kasėjai“ yra linkę pereiti prie sistemos, kadangi nėra pakankamų ekonominių paskatų, o pati sistema iš esmės neeliminuoja „frontrunning“ grėsmės, nei sumažina transakcijų vykdymo kaštus. Tai reiškia, kad sistema yra neefektyvi ir neveikianti pilnu pajėgumu. Lyginant su „replace-by-fee“ mechanizmu, pastarasis yra įdiegtas į „Ethereum“ protokolą ir jo vykdymas nepriklauso nuo kitų rinkos dalyvių, taip pagrįsdamas savo nepriklausomumo pranašumą.

Taigi, palyginus protokole esantį transakcijų rikiavimo mechanizmą su kitomis galimomis apsaugos galimybės, išryškėja „replace-by-fee“ privalumai prieš kitas siūlomas sistemas: užtikrinamas nepriklausomumas nuo kitų rinkos dalyvių ir gali būti vykdomas savarankiškai bei pasižymi dideliu vykdymo greičiu, kuris svarbus reaguojant į automatizuotos prekybos atakas.

2.5. Atakų nustatymo metodai

Siekiant išvystyti „frontrunning“ atakų prevencijos sistemą, svarbu pažymėti tai, kad nors tarpininkavimo pagrindas yra išmanioji sutartis, šis technologinis elementas nėra pajėgus įgyvendinti visus sistemos reikalavimus.

Visų pirma, išmaniųjų sutarčių vykdymas blokų grandinėje yra brangus ir ribotas. Pasak Eberhardt ir Tai (2017), blokų grandinėje vykdomi skaičiavimai turi ne tik aukštus kaštus, tačiau ir funkcinius apribojimus. Tai lemia, kad sudėtingų algoritmų vykdymas yra neįmanomas, kadangi pažeidžiama viršutinė sudėtingumo riba. Siekiant apeiti išmaniųjų sutarčių apribojimus dėl vykdymo kaštų ir funkcinių limtų, išeitis yra remtis už grandinės ribų vykdomais skaičiavimais.

Taip pat norint turėti funkcionalią prevencijos sistema, yra būtina pasiekti išorinius duomenis, tai yra sugebėti stebėti tinklą dėl tikėtinų atakų. Išmanieji kontraktai yra apriboti savybe – determinizmu. Dėl šios sąlygos išmaniosios sutartys negali vykdyti tinklo analizės, nes išoriniai duomenys yra kintantys, todėl tinklas negalėtų priimti sprendimo dėl vykdymo rezultato. (Eberhardt, Tai, 2017).

Dėl šių dviejų priežasčių – didelių skaičiavimo kaštų ir duomenų pasiekiamumo – išmaniosiose sutartyse vykdyti atakų analizę ne tik būtų ekonomiškai neefektyvu, tačiau yra ir techniškai neįmanoma. Kadangi „frontrunning“ atakų nustatymas yra paremtas nepatvirtintų transakcijų stebėjimu, o tai yra išoriniai tinklo duomenys, nepasiekiami iš išmaniųjų kontraktų, atakų nustatymas turi būti vykdomas ne grandinėje, o už jos ribų.

Kadangi optimalus sprendimas apima grandinėje vykdomus veiksmus (išmaniuosius kontraktus) ir ne grandinėje vykdomus analizės metodus, reikalingas metodas šiuos du sprendimus sujungti architektūriškai. Sistemą, susidedančią iš blokų grandinės lygio ir aplikacijos lygio, apibūdina Xu, Pautasso, Zhu ir Gramoll (2016). Tokioje sistemoje duomenų apsikeitimas vykdomas dviem būdais: aplikacija (sistemos dalis už grandinės ribų) komunikuoja su išmaniuoju kontraktu per transakcijas, o išmanieji kontraktai komunikuoja su aplikacija per nekintamą įvykių sąrašą, kurį seka aplikacija.

Taigi, sistemų atskyrimas tarp blokų grandinės ir aplikacijos lygių yra pagrįsta architektūra, siekiant apeiti išmaniųjų sutarčių apribojimus ir siekiant ekonominės naudos. Siekiant išlaikyti sistemą vientisą, komunikacija tarp skirtingų dalių vykdoma per blokų grandinės transakcijas arba įvykių pranešimus.

Transakcijų atpažinimas pagal požymius transakcijų sraute

Vienas iš lengviausiai apibrėžiamų metodų, skirtų nustatyti „frontrunning“ atakas, yra atrinkti atakos požymius tenkinančias nepatvirtintas transakcijas, kurių išskirtiniai bruožai yra identiški aukos transakcijos bruožams.

Pirmasis požymis, pagal kurį galima nustatyti, ar vykdoma ataka, pasak Qin, Zhou ir Gervais (2021), yra transakcijų, esančių atakoje, įskaitant užpuoliko ir aukos, buvimą tame pačiame bloke. Kadangi ataką bandome nuspėti dar prieš transakcijoms patenkant į bloką, vertinamos tik potencialios transakcijos, tai yra tokios transakcijos, kurios yra dar yra nepatvirtintos.

Antrasis požymis, nusakantis „frontrunning“ ataką, yra tų pačių žetonų pirkimas ta pačia kryptimi. Tai reiškia, kad užpuoliko ataka keis vienos rūšies žetoną į kitos rūšies žetoną lygiai taip pat, kaip tą daro auka. (Qin, Zhou, Gervais, 2021).

Trečiasis požymis nurodo, kad tiek užpuolikas, tiek auka naudojasi ta pačia decentralizuota birža. Todėl, jei vykdoma „frontrunning“ ataka, aukos ir užpuoliko transakcijos kreipsis į tą patį išmanųjį kontraktą (Qin, Zhou, Gervais, 2021).

Ketvirtasis požymis yra transakcijos vykdymo kaina. Šį požymį Torres, Camino (2021) apibūdina didesniu užpuoliko mokamu mokesčiu, negu auka, tam, kad „kasėjai“ būtų suinteresuoti ekonomiškai įtraukti puolančiąją transakciją į bloką prieš atakuojamą transakciją.

Apibendrinant transakcijų atpažinimo metodą pagal požymius nepatvirtintų transakcijų sraute, „frontrunning“ ataka gali būti atpažįstama nepatvirtintų transakcijų sraute remiantis keturiais bruožais ir juos lyginant aukos ir potencialaus užpuoliko transakcijas.

Išankstinė sandorių simuliacija ir poveikis kainai

Įvertinti, ar vykdoma ataka, nesiremiant „frontrunning“ atakos požymiais, yra bandyti simuliuoti sandorius dar prieš jiems realiai įvykstant, siekiant nustatyti, kaip pasikeis kaina, kai bus vykdomas aukos transakcijos sandoris.

Išankstinė sandorių simuliacija, anot Torres ir Camino (2021), yra metodas, skirtas išvengti aptikti transakcijas, kurios tenkina tik formalius „frontrunning“ atakos požymius, tačiau žalos nesukuria. Kad būtų sumažinta tikimybė gauti klaidingai teigiamus rezultatus pagal požymius, atakų paieškos metu vykdoma sandorių simuliacija. Simuliacijos tikslas, anot Qin, Zhou ir Gervais (2021), yra nustatyti, ar transakcija yra ne tik sėkminga, tačiau ir ar padaro reikšmingą poveikį biržai. Ataka pripažįstama simuliacijoje tik tokiu atveju, jei simuliacija įrodo, kad veiksmas atneša pelną užpuolikui arba, vertinant aukos požiūriu, jei nustatoma, kad užpuoliko transakcija reikšmingai paveiks turto kainą, dėl kurios bus sukurtas neigiamas praslydimas (angl. *slippage*) – skirtumas tarp kainos, už kurią buvo tikimasi įvykdyti sandorį, ir kainos, už kurią sandoris buvo iš tikrųjų įvykdytas.

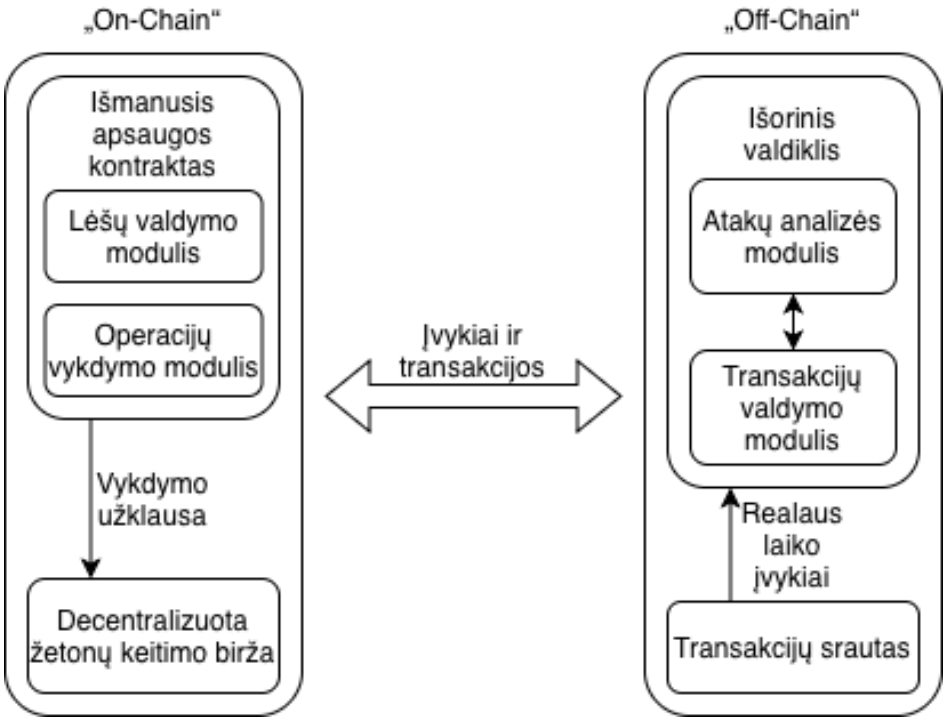
Apibendrinant simuliacijos metodą dėl poveikio kainai, pastarasis metodas gali būti naudojamas kaip atskiras metodas, bandant numatyti galimą ataką, bet ir kaip papildomas vertinimo

aspektas, kuris gali būti komponuojamas su atakų atpažinimu pagal atakos požymius, siekiant išvengti neteisingai teigiamų atakų įvertinimo.

2.6. Siūlomo sprendimo apžvalga

Išanalizavus sistemos poreikį, technines blokų grandinės ypatybes, susisteminus šiuo metu esančias apsaugojimo galimybes, įvardinus automatizuotos prekybos sukuriamą žalą ir sukategorizavus atakų nustatymų metodus, teikiamas siūlomas sprendimas, skirtas pasiūlyti naują decentralizuotų sandorių apsaugos sistemą, skirtą kovoti prieš rinkos manipuliacijas blokų grandinėje.

Įvertinus reikalingus siūlomos sistemos komponentus, atsižvelgiant į išnagrinėtas galimas technines priemones bei taikymo galimybes, projektuojama decentralizuotų sandorių apsaugos sistema, taikant transakcijų pakeitimo mokesčiu metodą. Aukšto lygio siūlomo sprendimo architektūra yra pateikiama 1 pav.



Šaltinis: sudaryta autoriaus, naudojantis draw.io įrankiu

1 pav. Siūlomo sprendimo aukšto lygio architektūros schema

Siūlomas sprendimas yra sudarytas iš dviejų dalių ir turėtų būti diegiamas dvejose aplinkose. Visų pirma, pasirinktoje blokų grandinėje, palaikančioje transakcijų pakeitimo mokesčiu mechanizmą (angl. *replace-by-fee*), diegiamas išmanusis kontraktas, turintis lėšų valdymo modulį, skirtą valdyti pinigines lėšas, kurios naudojamos vykdyti žetonų keitimo operacijas decentralizuotose biržose, bei operacijų vykdymo modulis, atliekantis pačią žetonų pakeitimo operaciją decentralizuotoje biržoje arba

lėšų grąžinimo operaciją, jei nustatoma ataka. Antroji siūlomos sistemos – išorinio valdiklio – dalis diegiama už grandinės ribų. Šis modulis nuolatos seka realiu laiku tinklo įvykius (transakcijų srautą), kad galėtų analizuoti vykdomas transakcijas, siekiant nustatyti galimas algoritminės prekybos atakas. Atakų analizė vykdoma išoriniame valdiklyje esančiame atakų analizės modulyje, kuris generuoja signalus transakcijų valdymo moduliui (gynybos mechanizmui), ar vykdyti žetonų keitimo ar inicijuoti transakcijos atšaukimo operaciją išmaniajame kontrakte.

Informacija tarp grandinėje (angl. *on-chain*) įdiegto išmaniojo kontrakto ir už grandinės ribų (angl. *off-chain*) esančiame išoriniame valdiklyje vykdoma skirtingais metodais, priklausančiais nuo žinučių perdavimo krypties. Išmanusis kontraktas informaciją su išoriniu valdikliu keičiasi remdamasis įvykiais (angl. *events*), o aplikacija (išorinis valdiklis) – vykdymas išmaniųjų kontraktų metodus transakcijose.

Atakų analizės modulis, diegiamas išoriniame valdiklyje, valdomas pasirinktu tinklo srauto analizės metodu. Siūlomame sprendime galimi du variantai: atakos transakcijų atpažinimas pagal transakcijų požymius ir išankstinės sandorių simuliacijos, siekiant nustatyti poveikį kainai, metodai. Atakų analizės modulyje veikiantis analizės metodas gali būti diegiamas pasirinktinai, netaikant apribojimų.

Siūlomo sprendimo – decentralizuotų sandorių apsaugos sistemos – pagrindinis veikimo principas yra išoriniame valdiklyje esantis transakcijų valdymo modulis. Sistemos naudotojų žetonų keitimo operacijos yra vykdomos išoriniam valdikliui kviečiant įdiegto išmanaus kontrakto, veikiančio tarpininko režimu, funkciją. Kadangi iškviestos funkcijos blokų grandinėje negali būti tiesiogiai atšaukiamos, naudojamas transakcijų pakeitimo mokesčiu metodas. Taikant šį metodą, pirkimo funkcija gali būti pakeista kita funkcija (siūlomo sprendimo atveju, žetonų grąžinimu), kad būtų atšauktas žetonų keitimas decentralizuotoje biržoje. Tokiu būdu, nustačius automatizuotos prekybos ataką, kuri gali sukelti žalą įprastam naudotojui, galima sustabdyti sandorius.

Apibendrinant siūlomą sprendimą, siekiama derinti blokų grandinės ypatumus ir už grandinės esančių sprendimų kombinaciją, siekiant pasiūlyti naują gynybos mechanizmą, siekiantį apsaugoti decentralizuotus sandorius nuo algoritminės prekybos.

Siūlomo sprendimo architektūra ir sistemos, skirtos apsaugoti decentralizuotus sandorius, esminės unikalios savybės yra blokų grandinėse jau esančių protokolo panaudojimas gynybai ir reaktyvios gynybos mechanizmo taikymas.

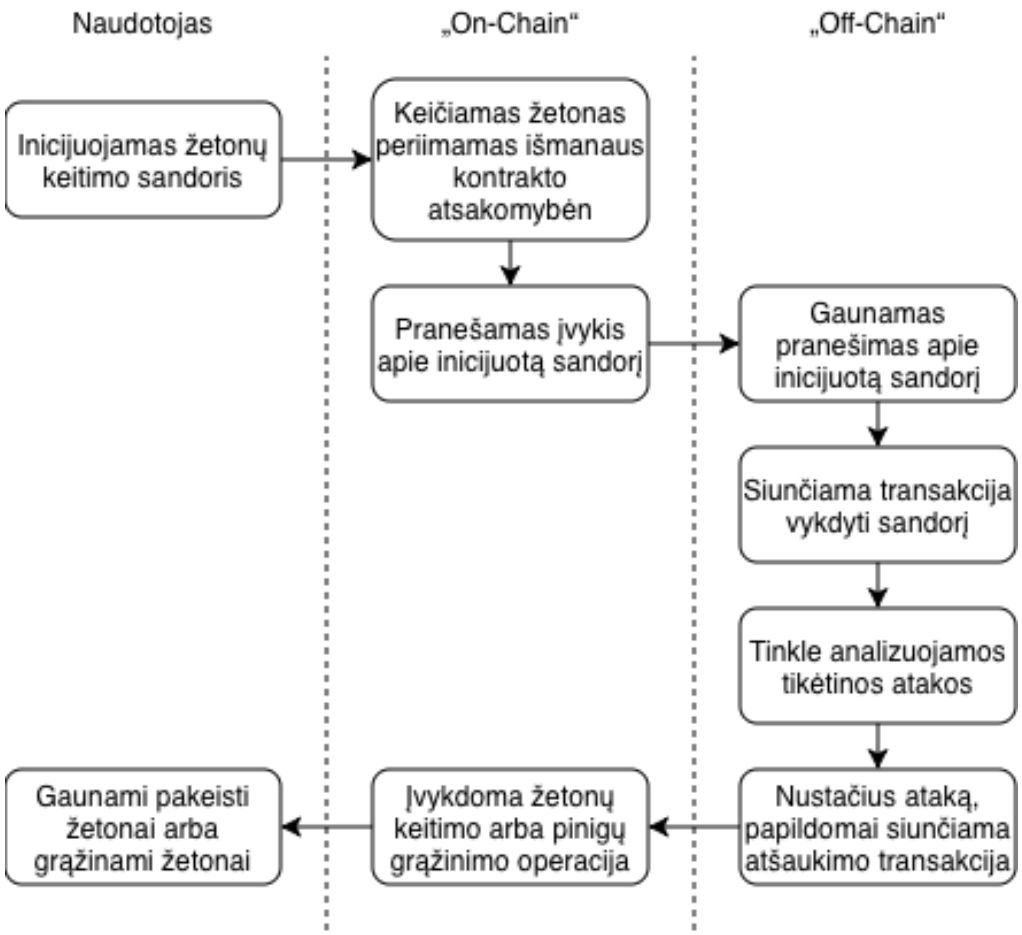
Visų pirma, transakcijų pakeitimo mokesčiu (angl. *replace-by-fee*) metodas yra jau egzistuojantis ir įdiegtas blokų grandinėse, tad siūlomas sprendimas nesiūlo naujos architektūros ar

protokolų, o išnaudoja jau esančią infrastruktūrą kitu nei numatyta būdu. Įprastai transakcijų pakeitimas mokesčiu yra taikomas siekiant pakoreguoti esamą nepatvirtintą transakciją, jei, pavyzdžiui, ši per ilgai yra neįtraukiama į bloką dėl per mažo mokesčio ar reikalingos didesnės transakcijos vykdymo spartos arba norint pataisyti klaidas, pavyzdžiui, neteisingai nurodžius gavėją. Siūlomo sprendimo atveju, šis protokolas išnaudojamas gynybai inicijuoti, siekiant visiškai pakeisti vykdomą išmanaus kontrakto metodą ir gauti skirtingą (rinkoje netikėtą) rezultatą.

Antra, sistema pasižymi savo realaus laiko stebėsenos funkcija. Tai reiškia, kad siūloma decentralizuotų sandorių apsaugos sistema yra aktyvus apsaugos mechanizmas, nustatantis ir neutralizuojantis atakas realiu laiku. Tai turi pranašumą prieš tokius apsaugos metodus, kaip „slippage“, kuri sprendžia atakų žalos padarinius, nei bando visiškai išvengti žalos susidarymo.

Apibendrinus sistemos unikalius bruožus – transakcijų pakeitimo mokesčiu metodo ir realaus laiko stebėsenos – sistema pasižymi savo naujumu ir pranašumu prieš egzistuojančius mechanizmus bruožu, nekuriant naujų technologinių sprendimų, išnaudoti egzistuojančius protokolus bei pasiūlyti realaus laiko sprendimą, kuris grindžiamas nuostolių išvengimu nei jų sušvelninimu.

Decentralizuotų sandorių apsaugos sistemoje veikia 3 dalyviai: naudotojas, „on-chain“ ir „off-chain“ sprendimo dalys. Decentralizuotų sandorių iniciatorius ir naudos gavėjas yra naudotojas, kuris su sistema sąveikauja vieną kartą, inicijuodamas sandorį. Blokų grandinėje esantis komponentas – išmanusis kontraktas – atlikdamas tarpininko vaidmenį, perima žetonų kontrolę ir praneša apie įvykį į už grandinės ribų esančią aplikaciją – išorinį valdiklį. Pastarojo komponento tikslas – pradėti vykdyti žetonų keitimo sandorį per tarpininką išmanųjį kontraktą, nuskaityti tinklą ir identifikuoti galimą ataką, o ją nustačius – inicijuoti gynybos mechanizmą ir vykdyti atšaukimo transakciją. Šis komponentas, realizuojantis gynybos mechanizmą, paremtą transakcijų pakeitimo mokesčiu metodu, yra pagrindinis siūlomo sprendimo komponentas. Atitinkamai sandorių keitimas vykdomas arba atšaukimas per išmaniojo kontrakto metodus, kurių rezultatas pakeistų žetonų pervedimas į naudotojo piniginę arba keičiamų žetonų grąžinimas. Siūlomo sprendimo veikimo žingsniai ir dalyvių atsakomybės pateikiamos 2 pav.



Šaltinis: sudaryta autoriaus, naudojantis draw.io įrankiu

2 pav. Aukšto lygio siūlomo sprendimo veikimo schema

Apibendrinant metodo veikimo žingsnius, naudotojas atlieka iniciatoriaus vaidmenį, išmanusis kontraktas saugiai įšaldo naudotojo lėšas ir veikia tarpininko režimu su decentralizuota birža, o pagrindinį darbą atlieka už grandinės ribų esantis išorinis valdiklis, kuris vykdo tinklo stebėseną ir priima sprendimą dėl sandorio vykdymo ar atšaukimo.

2.7. Efektyvumo vertinimo metodai

Siekiant įvertinti pasiūlytą apsaugos sistemą, bus siekiama nustatyti sprendimo efektyvumą. Šis efektyvumo tikslas padės įvertinti, ar sukurta decentralizuotų sandorių apsaugos sistema ir pasirinkti atakų aptikimo metodai yra pakankamai tikslūs, transakcijų pakeitimo mokesčiu metodas atlieka gynybinę funkciją ir efektyviai padeda išvengti automatizuotos prekybos atakų.

Pirmasis patikimumo aspektas yra saugumo efektyvumas. Šiuo rodikliu bus siekiama išsiaiškinti, ar sprendimas sėkmingai padeda apsisaugoti nuo atakos, kai ši vyksta. Šiuo rodikliu bandoma išsiaiškinti, ar sistema pakankamai greitai sureaguoja į kylančią grėsmę ir ar „replace-by-fee“

mechanizmas sėkmingai sustabdo ataką. Efektyvumas, kaip nusako Powers (2020), apibrėžiamas formule:

$$E = \frac{N_{apsaugota}}{N_{viso}},$$

čia E – efektyvumas, išreikštas procentais;

$N_{apsaugota}$ – apsaugotų transakcijų kiekis;

N_{viso} – visų transakcijų kiekis.

Antrasis patikimumo aspektas yra tikslumas. Šiuo rodikliu siekiama išsiaiškinti, kaip sistema tiksliai nustato atakas arba kiek atvejų sistema neatpažįsta realios atakos ir leidžia įvykti atakuojamam sandorio transakcijai. Tikslumas nusakomas formule, kurią apibrėžia Powers (2020):

$$T = \frac{TP}{TP+FP},$$

čia T – tikslumas;

TP – teisingai atpažintos atakos (angl. true positive);

FP – neteisingai identifikuotos atakos (angl. false positive).

Trečiasis patikimumo aspektas yra jautrumas. Šiuo rodikliu siekiama patikrinti, kokią dalį atakų sistema sugebėjo pastebėti. Aukštesnis įvertis parodys, kad tikimybė, jog naudotojas buvo sėkmingai atakuotas, yra mažesnė. Powers (2020) jautrumo rodiklį išreiškia formule:

$$J = \frac{TP}{TP+FN},$$

čia J – jautrumas;

TP – teisingai atpažintos atakos (angl. *true positive*);

FN – neatpažintos atakos (angl. *false negative*).

Vertinant pasiūlyto sprendimo efektyvumą, bus atskirai analizuojami atskiri – transakcijų atpažinimo pagal požymius ir išankstinės simuliacijos – atakų metodai, siekiant nustatyti, kuris iš pateiktų metodų galėtų būti patikimai naudojamas decentralizuotų sandorių apsaugos sistemoje. Atakų nustatymo metodai bus vertinami, remiantis tikslumo ir jautrumo aspektais, siekiant išanalizuoti šių metodų efektyvumą atakų atpažinime.

Taip pat atskirai vertinamas siūlomo sprendimo pagrindinis metodas, skirtas vykdyti gynybai – transakcijų pakeitimo mokesčiu metodas. Tam bus analizuojamas saugumo efektyvumas, kuris leis įvertinti, kaip efektyviai blokų grandinėje egzistuojantis protokolas gali būti panaudotas gynybos mechanizmuose.

Apibendrinant vertinimo kriterijus, atakų aptikimo metodai ir naudojamas gynybos metodas bus įvertinti atskirai, kad būtų galima įvertinti atskirus siūlomo sprendimo komponentus, bei įvertinti bendra visuma, kaip sistema, ar ji yra efektyvi mažinant automatizuotos prekybos atakas.

3. EKSPERIMENTINĖ APSAUGOS NUO AUTOMATIZUOTOS PREKYBOS ATAKŲ SISTEMA

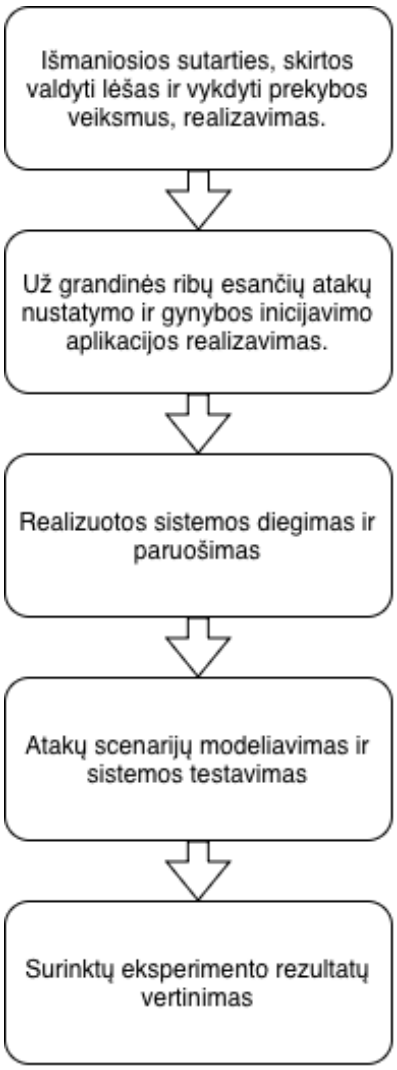
Šiame skyriuje pristatomas automatizuotų atakų prevencijos sistemos, paremtos „replace-by-fee“ metodu, projektavimas, eksperimentinio tyrimo metodika bei sprendimo patikimumo bandymas.

Eksperimento objektas

Eksperimento metu realizuojama prevencinė sistema, susidedanti iš dviejų dalių: išmaniosios sutarties, esančios blokų grandinėje ir atsakingos už lėšų perlaidos ir prekybos veiksmus, bei už grandinės ribų esančios aplikacijos, vykdančios tinklo analizę, siekiant nustatyti galimas automatizuotos prekybos strategijų atakas, ir inicijuojančios gynybos mechanizmą. Tyrimo metu nagrinėjama prekyba decentralizuotose biržose, kuriose vykdomi rinkos dalyvių žetonų keitimo sandoriai, kurie potencialiai tampa automatizuotų automatizuotos prekybos sistemų taikiniais.

Eksperimento eiga

Siekiant gauti ir įvertinti tyrimo rezultatus, išskiriami esminiai etapai: realizavimo, diegimo, paruošimo, testavimo ir vertinimo. Išsamūs eksperimento eigos žingsniai pateikti 3 pav.



Šaltinis: sudaryta autoriaus, naudojantis draw.io įrankiu

3 pav. Detalūs atliekamo eksperimento žingsniai

Siekiant užtikrinti artimiausią galimą simuliaciją „Ethereum“ tinklui, bus naudojamas „Sepolia“ tinklas, kuris leis patikrinti sistemos veikimą testinėje aplinkoje kuo artimesnėmis sąlygomis realiai blokų grandinei. Kadangi testavimas bus atliekamas su netikromis lėšomis, pradinis jų kilmės šaltinis testavime yra specialūs žetonų tiekimo šaltiniai – kranai (angl. *faucets*). Buvo naudoti du kranai žetonams gauti: „Google Cloud Web3 Ethereum Sepolia Faucet“ (adresu https://cloud.google.com/application/web3/faucet/ethereum/sepolia) ir „Sepolia PoW Faucet“ (adresu https://sepolia-faucet.pk910.de).

3.1. Sistemos realizavimas

Šiame skyrelyje pateikiamos išmaniosios sutarties, reikalingos tarpininko veiksmui atlikti, ir už grandinės ribų esančios aplikacijos – išorinio valdiklio – realizavimo ypatybės.

Išmaniosios sutarties realizavimas

Išmanioji sutartis, reikalinga lėšų valdymo ir prekybos tikslams, realizuojama „Solidity“ programavimo kalba, pasinaudojant kūrimo aplinkoje „Remix“. Išmanusis kontraktas apribotas naudoti naujausią „Solidity“ versiją (0.8.30), kodas kompiliuojamas naujausia galima versija (0.8.30). Išmaniojo kontrakto programinis kodas pateikiamas 1 priede.

Išmanaus kontrakto veikimą galima išskaidyti į dvi grupes pagal naudotojo grupes – savininko ir naudotojo. Savininkas kontraktu manipuliuoti galės naudojant dvi funkcijas – užsakymo vykdymo ar jo atšaukimo. Naudotojas galės naudoti dvi funkcijas – užsakymo pateikimo ir pinigų susigrąžinimo. Šios funkcijos pateikiamos 2 lentelėje.

2 lentelė

Išmanaus kontrakto funkcijų sąrašas

Funkcijos paskirtis	**Funkcijos pavadinimas**	**Aprašymas**
Užsakymo sukūrimas	placeOrder	Sukuria žetonų keitimo užsakymą, kurį reikia įvykdyti decentralizuotoje biržoje, bei perveda keičiamas lėšas į kontraktą
Pinigų grąžinimas	refund	Leidžia naudotojui atsiimti lėšas iš kontrakto už visus dar neužbaigtus užsakymus
Užsakymo vykdymas	executeOrder	Vykdo žetonų keitimo sandorį decentralizuotoje biržoje
Užsakymo atšaukimas	cancelOrder	Atšaukia užsakymą (nustačius ataką) ir grąžina užsakymo lėšas naudotojui

Šaltinis: sudaryta autoriaus

Lėšų valdymo operacijos išmaniajame kontrakte valdomas naudojant „ERC20“ sąsają, kurioje apibrėžiamos trys funkcijos: pinigų pervedimo iš nurodyto adreso (šio kontrakto atveju, iš naudotojo), kontrakto vykdymo pinigų pervedimo bei lėšų teisių suteikimo. Žetonų keitimas vykdomas decentralizuotoje biržoje vykdomas naudojant „UniswapV2Router02“ sąsają, kurioje naudojama vienintelė funkcija, skirta pakeisti visą suteiktą žetonų kiekį į pasirinktus žetonus.

Išmanaus kontrakto panaudojimas apsaugos sistemoje atliekamas dvejais etapais: užsakymo sukūrimo ir vykdymo.

Užsakymo pateikimas susideda iš parametrų surinkimo iš naudotojo apie ketinimą vykdyti žetonų keitimą (decentralizuotos biržos adreso, parduodamo ir perkamo žetono adresų, parduodamo žetono kiekio, tikėtino mažiausio perkamo žetonų kiekio bei laiko žymos, iki kada užsakymas galioja).

Šio metodo vykdymo metu atliekama lėšų pervedimo operacija iš naudotojo piniginės į išmanųjį adresą. Sėkmingai atlikus lėšų pervedimą ir užsakymo duomenų išsaugojimą, tinklas informuojamas sukurto užsakymo įvykiu.

Sukurtas užsakymas pakeisti žetonas decentralizuotoje biržoje yra apdorojamas už grandinės ribų esančioje aplikacijoje, kai gaunamas sukurto užsakymo įvykis. Analizės ir transakcijų vykdymo sistemos dalis inicijuoja žetonų keitimo operaciją kviečiant išmaniojo kontrakto metodą, o atlikus tinklo analizę ir nustačius galimą ataką, atšaukia vykdymą, pakeičiant vykdomą metodą į atšaukimo funkciją, ir įvykdo užsakymo atšaukimo bei lėšų grąžinimo operaciją.

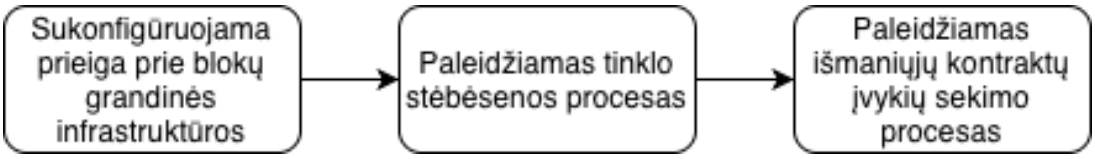
Siekiant užtikrinti lėšų saugumą, naudotojas, pateikęs užsakymus, juos gali atšaukti, jei šie nebuvo įvykdyti, ir susigrąžinti visas į išmanųjį kontraktą pervestus žetonus.

Už grandinės ribų esančios aplikacijos realizavimas

Už grandinės ribų esanti aplikacija, atsakinga už žetonų keitimo užsakymo įvykdymą ir atakų nustatymą, yra realizuojama C# programavimo kalba ir .NET karkasu, kurios versija yra naujausia (9.0). Realizacija atlikta naudojantis „JetBrains Rider“ kūrimo aplinka. Aplikacijos, esančios už grandinės ribų, programinis kodas yra pateikiamas 2 priede.

Sąveika su blokų grandine („Sepolia“) užtikrinama naudojant atviro kodo biblioteką „Nethereum“ (versija 5.0.0). Ši biblioteka leidžia programiškai kviesti išmaniojo kontrakto funkcijas, pasirašyti transakcijas. Prieiga prie blokų grandinės tinklo prieinama per „Infura“ paslaugų tiekėją, leidžiantį realiu laiku stebėti tinkle vykdomas transakcijas.

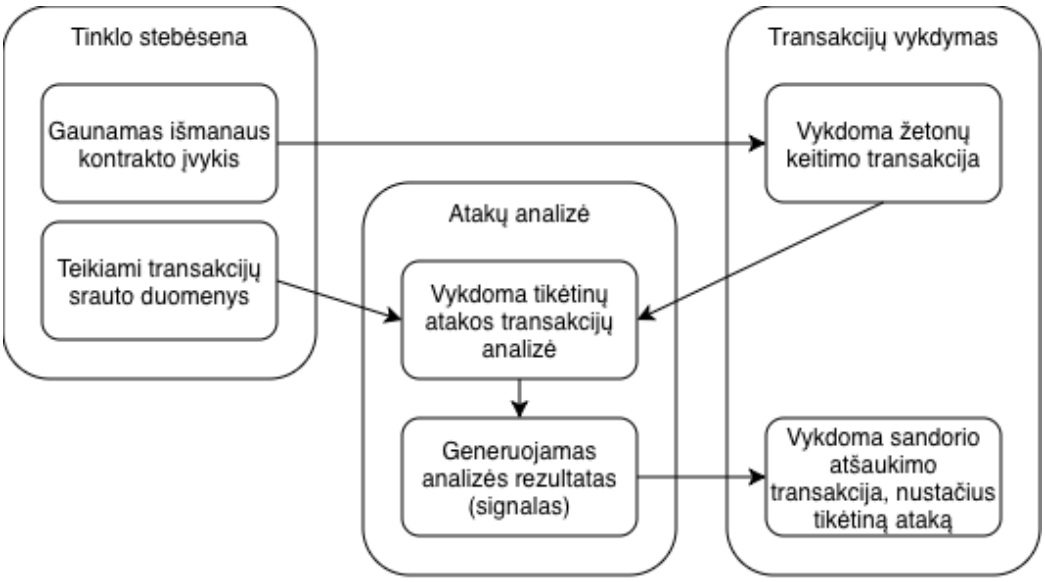
Išorinis valdiklis, programos įeigos metu, atlieka tris žingsnius: sukonfigūruoja prieigą prie infrastruktūros, paleidžia foninius procesus, skirtus tinklo stebėsenai ir įvykių sekimui. 4 pav. pavaizduota schema, nusakanti programos įeigos žingsnius.



Šaltinis: sudaryta autoriaus, naudojantis draw.io įrankiu

4 pav. Išorinio valdiklio programos įeigos žingsniai

Šios sistemos dalies realizacija leidžia sekti realiu laiku sekti išmaniojo kontrakto įvykius (žetonų keitimo užsakymo pateikimo) ir atitinkamai, nustačius galimą ataką, aktyvuoti gynybos mechanizmą, paremtą transakcijos pakeitimą mokesčiu. Komponentų priklausomybės ir veikimo eigos principas parodytas 5 pav. esančioje schemoje.



Šaltinis: sudaryta autoriaus, naudojantis draw.io įrankiu

5 pav. Išorinio valdiklio komponentų eigos ir priklausomybių schema

Tolimesniame poskyryje aprašomas realizuotos sistemos diegimas ir paruošimas veikimui bei testavimui.

3.2. Realizuotos sistemos diegimas ir paruošimas

Siekiant eksperimentiškai patikrinti įgyvendintą decentralizuotų sandorių apsaugos sistemos prototipą nuo automatizuotos prekybos atakų, buvo sukurtos dalyvių piniginės (įprasto naudotojo ar savininko ir atakuotojo), įdiegtas išmanusis kontraktas „Sepolia“ tinkle bei paruošti žetonai prekybai.

Visų pirma, sukurtos dvi atskiros piniginės skirtingiems rinkos dalyviams – įprastam naudotojui (savininkui, sistemos naudotojui) bei atakuotojui. Įprastas naudotojas – išmanaus kontrakto savininkas – pinigine naudojasi siekiant valdyti išmanųjį kontraktą (pasinaudojant išoriniu valdikliu, vykdyti žetonų keitimo operacijas decentralizuotose biržose, atlikti gynybinius veiksmus) bei inicijuojant žetonų keitimo sandorius. Atakuotojo piniginės rolė eksperimente yra atlikti atakų transakcijas, nutaikytas prieš įprasto naudotojo žetonų keitimo sandorius. Apibendrinta piniginių informacija su adresais „Sepolia“ tinkle pateikiama 3 lentelėje. Šios piniginės buvo papildytos 1 ETH, o vėliau jų balansas papildomai papildytas tik esant poreikiui vykdyti transakcijas.

3 lentelė

Eksperimente naudotos piniginės, adresai ir jų paskirtis

Dalyvis	**Adresas**	**Paskirtis**
Savininkas	0xe14e1aF8bf9A98D888c363d905dA2FD3c83cdACD	Įdiegti išmaniąją sutartį blokų grandinėje ir valdyti išmanųjį kontraktą iš išorinio valdiklio, vykdyti žetonų keitimo operacijas per išmanųjį kontraktą.
Atakuotojas	0x5A439Fae39528f6cC27f60D841F3104a2B5409A2	Atlikti atakų transakcijas, nutaikytas į savininko sandorius

Šaltinis: sudaryta autoriaus

Turint sukurtas pinigines, pasinaudojant savininko pinigine, į „Sepolia“ tinklą įdiegiamas realizuotas išmanusis kontraktas, kuris veiks kaip tarpininkas vykdyti žetonų pirkimo sandorius decentralizuotose biržose. Išmanaus kontrakto diegimo informacija pateikta 4 lentelėje.

4 lentelė

Išmanaus kontrakto diegimo „Sepolia“ tinkle transakcijos informacija

Transakcijos maiša	0xc5e44dbe22d4ebfdfb4096dc1994a2476df26697ac82aad11ec641b49f872507
Išmanaus kontrakto adresas	0x6aE58F88d297F060E39783069A03325d1231bCA6
Transakcijos kaina	0.001933596011601576 ETH

Šaltinis: sudaryta autoriaus pagal sepolia.etherscan.io informaciją

Kadangi decentralizuotos biržos keičia skirtingų rūšių žetonus, eksperimente pasirinkta naudoti UNI ir WETH žetonai. Pastarasis žetonas yra ETH atitikmuo, tačiau laikomas „įvyniotu“ (angl. *wrapped*) į ERC20 žetono sąsają. Šie žetonai, naudojami decentralizuotose biržose, turi savo adresus, kurie pateikti 5 lentelėje. Atitinkamai, prieš vykdant eksperimentą, dalis pradinio ETH, esančio piniginėse, yra „įvyniojamas“ į WETH žetoną, decentralizuotose keityklose pusė lėšų pakeičiama į UNI žetoną toje pačioje piniginėje, kad būtų paruošta atlikti skirtingų krypčių žetonų keitimus.

5 lentelė

Eksperimente naudotų esamų ERC20 žetonų adresai

ERC20 žetonas	Žetono adresas
WETH	0xfFf9976782d46CC05630D1f6eBAb18b2324d6B14
UNI	0x1f9840a85d5aF5bf1D1762F925BDADdC4201F984

Šaltinis: sudaryta autoriaus pagal sepolia.etherscan.io informaciją

Pagal ERC20 sąsaja, naudojama išmaniuosiuose kontraktuose, šie negali disponuoti naudotojų žetonais tol, kol jie nesuteikia reikalingų leidimų, tai yra neatlieka patvirtinimo veiksmo (angl. *approve*). Kiekviena piniginė, naudojama eksperimente, priklausomai nuo rūšies, turi suteikti kiekvienam iš šių žetonų atitinkamus sutikimus: naudotojo piniginė – įdiegtam išmaniajam sistemos kontraktui, atakuotojas – tiesiogiai decentralizuotos biržos išmaniajam kontraktui. Remiantis galiojančia „Uniswap“ dokumentacija, decentralizuotos biržos išmanaus kontrakto adresas „Sepolia“ tinkle yra 0xeE567Fe1712Faf6149d80dA1E6934E354124CfE3, remiantis https://docs.uniswap.org/contracts/v2/reference/smart-contracts/v2-deployments. Žetonų naudojimo patvirtinimo transakcijos yra pateikiamos 6 lentelėje.

6 lentelė

Žetonų disponavimo išmaniuosiuose kontraktuose patvirtinimo transakcijos

Piniginė	Žetonas	Transakcijos maiša
Naudotojo	WETH	0xe98ae0159109fc85d76d2978770441d30a36df6f2e3945541b7485e935bef0ee
Naudotojo	UNI	0x7e859335cf429af153e55ef10891188dac0c984cbd1533790510b2cd9f3a27e0
Atakuotojo	WETH	0xed5ec45ba775c240cf10d3ca2fe1c131480c03becbf420562b06d03a6e27871b
Atakuotojo	UNI	0x27349f7adaca5d03d1602c4cee27eff377c916a2470fc9e7b900455e9517f3bb

Šaltinis: sudaryta autoriaus pagal sepolia.etherscan.io informaciją

Vykdant eksperimentinį testavimą su atakų nustatymo metodu, kuris remiasi transakcijos požymiais, užtenka naudoti jau „Sepolia“ tinkle esančius žetonus, tačiau simuliacijos metodui egzistuojantys žetonai netinka, nes decentralizuotose biržose esantys žetonų keitimo baseinai (angl. *pools*), kurie suteikia žetonams likvidumą, yra labai dideli, tad sunku sudaryti kainos pokytį decentralizuotoje biržose su turimais testavimo žetonais. Dėl šios priežasties reikalinga sukurti naują

ERC20 žetoną, skirtą simuliacijos metodui įvertinti, jį įdiegti „Sepolia“ tinkle bei sukurti likvidumą decentralizuotoje biržoje.

Reikalingo ERC20 žetono, pavadinto „Magistras2025“ su simboliu „MBD25“, diegimo informacija pateikta 7 lentelėje, o išmaniojo kontrakto programinis kodas – 3 priede.

7 lentelė

ERC20 žetono kūrimo „Sepolia“ tinkle transakcijos informacija

Transakcijos maiša	0x2e22c3f6e8d0b5ac5912e805d4f662a9f462b8acc01c15526944ebe2541bbd55
Išmanaus kontrakto adresas	0xe845444074c556aa351faf8478f529d95db2d956
Transakcijos kaina	0.00142216503128763 ETH

Šaltinis: sudaryta autoriaus pagal sepolia.etherscan.io informaciją

Įdiegus žetoną, reikia sukurti jo likvidumą decentralizuotoje biržoje. Tai atliekama sudarant tokią poziciją tarp naujo žetono „MBD25“ ir „WETH“, kad jų kaina pradinėje būsenoje būti lygi vienas prie vieno. Pozicijos sukūrimo informacija pateikta 8 lentelėje. Viso eksperimento metu, po kiekvieno simuliuojamo eksperimento scenarijaus, reikės užtikrinti, jog kaina decentralizuotoje biržoje yra kuo artesnė santykiui vienas prie vieno.

8 lentelė

Žetono likvidumo sudarymo decentralizuotojo biržoje transakcijos informacija

Transakcijos maiša	0x2e22c3f6e8d0b5ac5912e805d4f662a9f462b8acc01c15526944ebe2541bbd55
Išmanaus kontrakto adresas	0xe845444074c556aa351faf8478f529d95db2d956
Transakcijos kaina	0.00142216503128763 ETH

Šaltinis: sudaryta autoriaus pagal sepolia.etherscan.io informaciją

Kaip ir taikyta jau esamiems „Sepolia“ tinkle žetonams, reikalingas suteikti disponavimo teises tiek išmaniajam kontraktui, tiek decentralizuotai biržai naujai sukurtam žetonui, skirtam įgyvendinti išankstinės simuliacijos metodą nustatant atakas. Įprasto naudotojo piniginė suteikia disponuoti žetonu išmaniajam kontraktui (transakcijos maiša 0xa0202d20dbac880e5b3fcc4381f1a348c8d0443b0fb1f898edd57d448c4797b9), o atakuotojo piniginė

suteikia tiesiogiai decentralizuotai biržai (transakcijos maiša 0x6396d6167dae725d44d241fd2a260962dcfc65516ed5c8ef9fe0a5f5e04b3cd8).

3.3. Atakų scenarijų modeliavimas

Siekiant patikrinti sistemos veikimą įvairiomis situacijomis, sudaromas eksperimentinis duomenų rinkinys su trijų rūšių scenarijais: įprasti žetonų keitimo sandoriai, kurie vykdomi sąžiningai ir nėra atakuojami, scenarijai su atakuojančiomis transakcijomis, kurios vykdo tų pačių žetonų keitimo operaciją tose pačiose biržose mokant didesnį transakcijos mokestį, bei sandoriai, ne visai atitinkantys atakų požymius, sukuriant ribines situacijas.

Sudarant atakų scenarijų rinkinius, numatomas pasiskirstymas yra 30 scenarijų, iš kurių 12 scenarijų skirta pagrindimui, kad sistema netrikdo įprasto žetonų keitimo sandorių, 12 atakų scenarijų – atakų vertinimui, kuriais bus bandoma patikrinti, ar sistema gina sandorius nuo atakų, bei 6 scenarijai – ribiniai scenarijai, kurie turi atakos požymių, tačiau nekelia grėsmės rinkos manipuliacijai.

Eksperimentinėje scenarijų grupėje, susidarančioje iš įprastų sandorių ir atakų scenarijų, apibrėžiami esminiai sistemos veikimo modeliai. Įprasti sandoriai imituoja standartinį naudotojo elgesį, kai už operacijas mokama standartinė, tuo metu galiojanti rinkos kaina, keičiamų žetonų kiekis yra nereikšmingas ir nedarantis stiprių rinkos pokyčių, todėl tokios transakcijos, sistemos vertinimu, turėtų būti sėkmingai įvykdomos. Atakų scenarijuose – transakcijų mokestis padidinamas aukščiau rinkos lygio (bent penkiolika procentų), keičiama žetonų vertė yra pakankamai reikšminga, kad paveiktų žetono vertę biržoje, o toks derinys sistemai turėtų signalizuoti, kad vykdoma ataka ir privaloma naudoti gynybos mechanizmą.

Siekiant sudaryti ribinių situacijų, modeliuojamos tokios atakos, kurios tik iš dalies atitinka atakos apibrėžimą: scenarijuose modeliuojama neatitinkanti vieno iš bruožo, reikalingo sėkmingam identifikavimui, ataka. Pavyzdžiui, tokiame scenarijuje atakuojanti transakcija neturės vieno iš charakteristikų: identiškų keičiamų žetonų, lyginant su atakuojama transakcija, didesnės nei rinkos transakcijos mokesčio arba santykinai mažo keičiamo žetonų kiekio. Šie scenarijai leis įvertinti kraštutines atakų aptikimo metodų ribas nustatant atakas.

Eksperimento metu modeliuojamų scenarijų charakteristikos ir tikslas apibendrintai pateikiami 9 lentelėje.

9 lentelė

Eksperimente modeliuojamų atakų scenarijų charakteristikos ir tikslai

Tipas	Kiekis	Charakteristikos	Tikslas
Įprasti sandoriai	12	Taikomi standartiniai rinkos mokesčiai, keičiamas nereikšmingas žetonų kiekis be stiprios įtakos rinkai.	Patikrinti, ar sistema nebando stabdyti įprastų keitimo sandorių
Atakų scenarijai	12	Mokami didesni nei rinkos mokesčiai už transakcijas, keičiamas didesnis žetonų kiekis, keičiami žetonai atitinka aukos keičiamus žetonus	Nustatyti, ar sistema sėkmingai identifikuoja atakas ir jas neutralizuoja
Ribinės situacijos	6	Taikomi žemesni arba įprasti mokesčiai už transakcijas, keičiamas kiekis gali būti mažas arba panašus į aukos, keičiami žetonai gali skirtis	Patikrinti, kaip sėkmingai sistema išvengia klaidingai teigiamų atakų nustatymų

Šaltinis: sudaryta autoriaus

Kiekvienam iš modeliuotų scenarijų, nepriklausomai nuo jų tipo, parenkami trijų rūšių parametrai: žetonų keitimo krypties, dujų kainos koeficientas ir keičiamų žetono kiekio koeficientas. Pirmajam parametrui – žetonų keitimo krypties – atakos modeliuojamos taip, kad keičiamų žetonų kryptis sutaptų arba būtų priešinga naudotojo atliekamam sandoriui. Antrasis dujų kainos koeficientas nusako daugiklį, keliais kartais atakos transakcijos kaina bus didesnė nei naudotojo. Scenarijuose galimos koeficiento reikšmės siekia nuo 0,5 iki 6 kartų. Trečiuoju parametru nusakomas kiekis kartais, kiek daugiau atakos transakcija keis žetonų decentralizuotoje biržoje. Galimas daugiklis, modeliuojant scenarijus, yra tarp 0,001 ir 10 kartų. Detalios reikšmės kiekvienam eksperimente modeliuotam scenarijui pateikiamos 4 priede.

3.4. Sistemos testavimas

Decentralizuotų sandorių apsaugos sistemos prototipo transakcijų vykdymo modulis ir atakų nustatymo moduliai veikė viso eksperimento metu fiksuotais, bet, esant poreikiui, keičiamais parametrais.

Transakcijų vykdymo modulis, teikdamas tinklo stebėsenos duomenis į atakų analizės modulius, laukdavo ir rinkdavo tinklo duomenis už praėjusias keturias sekundes po žetonų keitimo sandorio transakcijos paskelbimo. Realiu laiku surinkta tinkle pastebėtų transakcijų informacija į atakų analizės modulius būdavo perduodama nufiltravus pašalines, ne iš eksperimento metu apibrėžtų piniginių adresų, vykdytas transakcijas, jog būtų sumažintas „triukšmas“, galėjęs paveikti eksperimento rezultatus.

Atakų nustatymo pagal transakcijų požymius metode buvo taikomas dujų kainos padidėjimo parametras – 10 procentų. Tai reiškia, kad lyginant tikėtinas tinkle matomas atakų transakcijas su naudotojo transakcija, kad ataka būtų nustatyta, reikėjo, jog mokama kaina viršytų pasirinktą dešimties procentų padidėjimo ribą. Taip pat, visos naudotojo transakcijos buvo atliekamos identiškais parametrais, tai yra inicijuojant žetonų keitimo sandorį pakeisti 0,001 WETH žetoną į UNI žetoną.

Išankstinių sandorių simuliacijos ir poveikio kainai metode buvo taikomas trijų procentų rinkos kainos pasikeitimo parametras. Transakcijos, kurių sandoriuose keičiamų žetonų kiekis kainą decentralizuotoje biržoje paveikdavo daugiau nei pasirinktas parametras (3 procentai), būdavo traktuojamos kaip atakos. Naudotojo transakcijos, vykdytos šio metodo eksperimento metu, buvo identiškos: inicijuojamas žetonų keitimo sandoris visada buvo sudarytas iš 0,0001 WETH žetono keitimo į sukurtą žetoną MBD15. Taikyti konfigūruojami parametrai eksperimento metu pateikti 10 lentelėje.

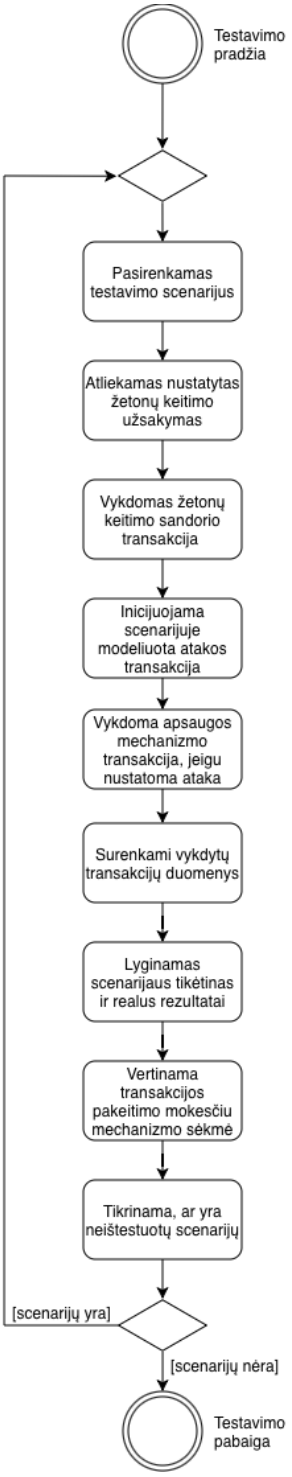
10 lentelė

Eksperimento metu taikyti konfigūruojami parametrai

Parametras	**Taikyta sąlyga**
Transakcijų analizės periodas	4 sekundės
Transakcijų filtravimas	Iš transakcijų srauto išfiltruojamos nesusijusių pinigių transakcijos
Atakos transakcijos kainos padidėjimas neklasifikuojant atakos	Ne daugiau 10 procentų
Maksimalus rinkos kainos poveikis neklasifikuojant atakos	Ne daugiau 3 procentų
Atakų atpažinimo pagal transakcijos požymius metode taikytos naudotojo transakcijos	0,001 WETH žetono keitimas į UNI žetoną
Išankstinių sandorių metode taikytos naudotojo transakcijos	0,0001 WETH žetono keitimas į sukurtą MBD15 žetoną

Šaltinis: sudaryta autoriaus

Eksperimento testavimui naudotas algoritmas pateikiamas 6 pav., kuriame nurodomi žingsniai, reikalingi testavimo scenarijams inicijuoti, vykdyti ir duomenims surinkti.



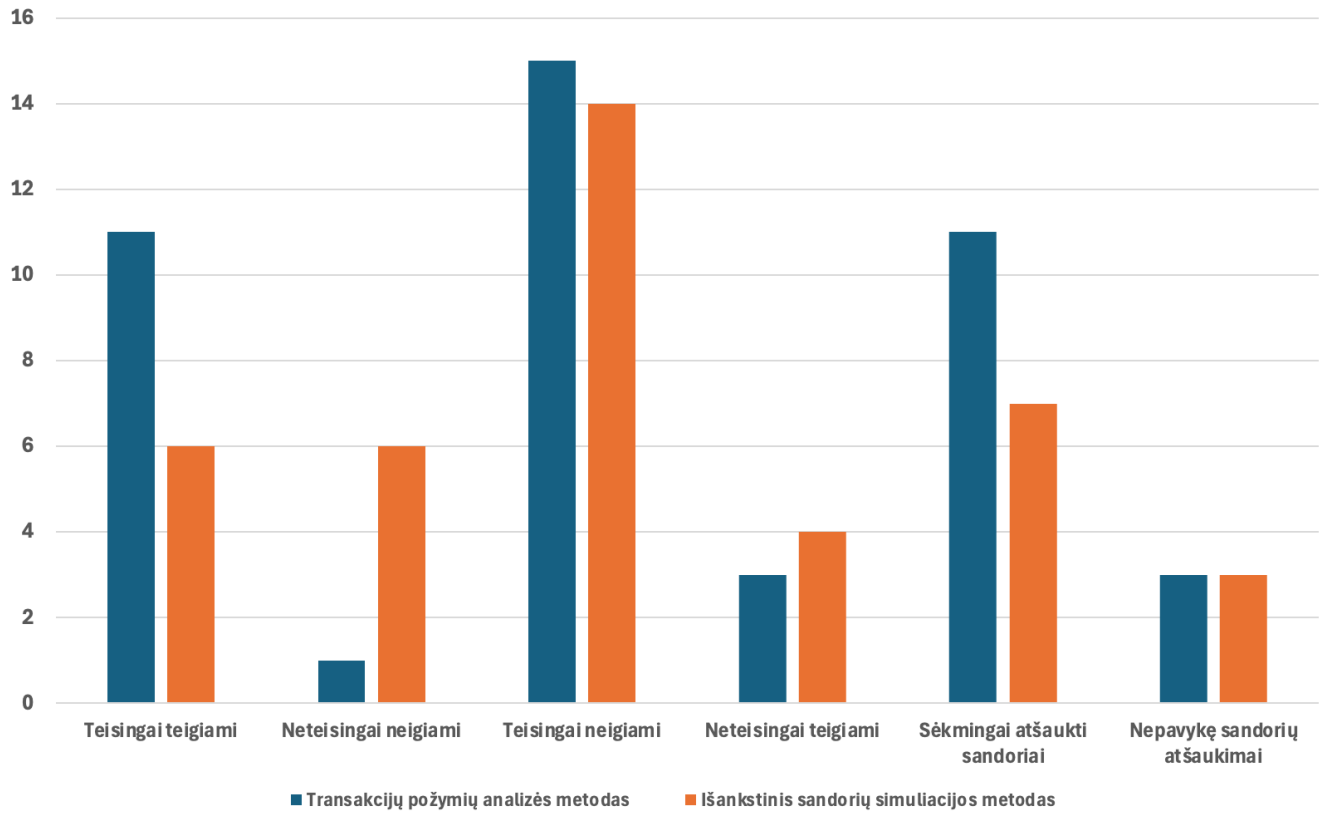
Šaltinis: sudaryta autoriaus, naudojantis draw.io įrankiu

6 pav. Eksperimento metu vykdyto testavimo algoritmo schema

Eksperimentas atliktas atskirai kiekvienam atakų atpažinimo metodui su identiškais scenarijais: kiekvienu atveju taikyti 30 modeliuotų scenarijų, iš kurių 12 – atakų transakcijų, 12 – įprastų

sandorių ir 6 ribiniai scenarijai. Įvykdytų transakcijų informacija atakų nustatymo metodo pagal transakcijų požymius pateikta 5 priede, išankstinių sandorių simuliacijos metodo – 6 priede.

Eksperimento metu surinkti duomenys kiekvienu atakų nustatymo metodo atveju pateikiami 7 pav.



Šaltinis: sudaryta autoriaus „Excel“ programoje

7 pav. Suklasifiktuotų scenarijų atvejų rezultatų grafikas

Remiantis surinktais duomenimis, atakų nustatymo pagal transakcijų požymius metode nustatyta teisingai teigiamų buvo 11, teisingai neigiamų – 15, neteisingai teigiamų – 3, neteisingai neigiamų – 1 atvejis. Tarp šių atvejų buvo 14 bandymų atšaukti žetonų keitimo sandorio transakcijas, kuriose sėkmingai buvo atšaukta 11, nepavyko – 3. Pagal surinktus duomenis, metodo efektyvumo vertinimo kriterijai apskaičiuojami taip:

$$E = \frac{11}{14} \approx 0{,}786,$$

$$T = \frac{11}{11+3} \approx 0{,}786,$$

$$J = \frac{11}{11+1} \approx 0{,}917.$$

Išankstinių sandorių simuliacijos metode, pagal surinktus duomenis, nustatyta teisingai teigiamų atvejų – 6, teisingai neigiamų – 14, neteisingai teigiamų – 4, neteisingai neigiamų – 6 atvejais.

Eksperimento metu buvo bandoma atšaukti 10 žetonų keitimo vykdymo operacijų, iš jų 7 buvo sėkmingos, o 3 – nesėkmingos. Pagal surinktus eksperimento duomenis, efektyvumo vertinimo kriterijai skaičiuojami taip:

$$E = \frac{7}{10} = 0{,}7,$$

$$T = \frac{6}{6+4} \approx 0{,}6,$$

$$J = \frac{6}{6+6} = 0{,}5.$$

Saugumo efektyvumas gali būti apjungiamas ir skaičiuojamas bendrai tarp šių metodų. Bendruoju atveju buvo 24 atvejai, kuomet buvo taikytas transakcijos pakeitimo mokesčiu metodas, iš jų 18 sėkmingai buvo įvykdyti, o nesėkmingai 6. Saugumo efektyvumo rodiklis šiam visam atvejui apskaičiuojamas taip:

$$E = \frac{18}{24} = 0{,}75.$$

Tolimesniame poskyryje pateikiamas eksperimento metu surinktų duomenų vertinimas.

3.5. Rezultatų vertinimas

Remiantis atliktais efektyvumo vertinimo metodų skaičiavimais, galima pastebėti, jog transakcijų atpažinimo pagal požymius metodas eksperimento sąlygomis yra pranašesnis prieš išankstinių sandorių simuliacijos metodą.

Vertinant metodų tikslumo kriterijų, kuris nusako, kiek iš nustatytų atakų buvo tikros, pastebimas atakų pagal transakcijų požymius metodo pranašumas (lyginant 78,6% su 60%). Šis įvertis rodo, kad požymiais grindžiamas atakų nustatymo metodas turėjo mažiau klaidingų signalų, o todėl įprastų naudotojų sandorius sutrikdė rečiau.

Lyginant jautrumo kriterijų, vertinantį kokią dalį atakų sistema sugebėjo pastebėti, nustatyta, kad išankstinių simuliacijų metodas fiksavo tik pusę atakų, kuomet transakcijų požymiais grindžiamas metodas pademonstravo aukštą – 91,7 procento – jautrumą.

Eksperimento sąlygomis patikrintas transakcijų pakeitimo mokesčiu metodas (angl. *replace-by-fee*) sėkmingas bendra apimtimi, vertinant abiejų atakų atpažinimo metodus, buvo 75 procentai (transakcijų požymių metode – 78,6 procento, išankstinių sandorių metode – 50 procentų). Lyginant su mokslinėje literatūroje minimu 95 procentų sekmės rodikliu, šio minimo lūkesčio eksperimento metu nepavyko pateisinti. Rezultatas rodo, kad decentralizuotų sandorių apsaugos sistemos konceptas yra veikiantis, bet nepakankamas naudojimui realiomis sąlygomis.

Esminis pasiūlyto sprendimo privalumas, atsižvelgiant į surinktus duomenis ir rezultatus, yra aukštas sistemos jautrumas, jei pasirenkamas transakcijų požymių analize grindžiamas atakų nustatymo metodas. Eksperimento rezultatai taip pat patvirtina ir decentralizuotos apsaugos, pagrįstos transakcijų pakeitimu mokesčiu, pakankamai sėkmingu veikimu, o pasiektas 75% rodiklis rodo, kad yra šio protokolo pritaikymo galimybės gynybos mechanizmų kūrimui.

Atsižvelgiant į sistemos apribojimus, nustatytas 20% atotrūkis tarp teorinio ir praktinio transakcijų pakeitimo mokesčiu lūkesčio. Šį skirtumą, tikėtina, galima paaiškinti techniniais tinklo pralaidumo ypatumais, kuomet pakeičianti transakcija nespėja būti įtraukta į bloką. Taip pat įžvelgiamas ir fiksuotų parametrų trūkumas: atakų nustatymo metoduose šie eksperimento metu veikė griežtai apibrėžtomis sąlygomis, kurios galėjo iškraipyti rezultatus.

IŠVADOS

1. Išanalizavus blokų grandinės veikimo principus ir išmaniųjų kontraktų galimybes, kuriant automatizuotus prevencijos mechanizmus, nustatyta, kad nors viešas duomenų prieinamumas blokų grandinėje užtikrina sistemos skaidrumą, tačiau išmanieji kontraktai yra apriboti savo nepakeičiamumu, kuris sumažina algoritmų lankstumą, ir didele vykdymo kaina, kuri sprendimus padaro ekonomiškai neefektyvius, todėl tikslinga taikyti hibridinius sprendimus, apimančius sistemos išskaidymą į blokų grandinėje ir už jos ribos veikiančias aplikacijų dalis, siekiant apeiti technines kliūtis.
2. Formalizavus automatizuotas prekybos sistemas, nustatyta, kad nors kai kurios algoritminės strategijos ir padeda rinkoms užtikrinti likvidumą, egzistuoja maksimaliai išgaunamos vertės atakos, kurios, pasinaudojant galimybe stebėti nepatvirtintas transakcijas realiu laiku bloke ir perrikiuoti transakcijų eiliškumą bloke, sukelia tiesioginę finansinę žalą rinkos dalyviams, manipuliuojant turto kainomis.
3. Pasiūlytas automatizuotas apsaugos metodas yra grindžiamas hibridine architektūra, kuriame decentralizuoti sandoriai vykdomi grandinėje įdiegtame išmaniajame kontrakte, atliekančiame tarpininko funkciją tarp naudotojo ir decentralizuotos biržos, ir už grandinės ribų esanti aplikacija vykdo tinklo analizę, siekiant nustatyti atakas, o jas identifikavus – taikomas transakcijų pakeitimas mokesčiu metodas, siekiant apsaugoti naudotoją nuo galimos automatizuotos prekybos sukeliančios žalos.
4. Remiantis nustatytais reikalavimais, realizuotas apsaugos sistemos prototipas, kurį sudaro „Sepolia“ testiniame tinkle įdiegtas išmanusis kontraktas ir C# programavimo kalba parašyta už grandinės ribų esanti aplikacija, kuri, naudodama „Nethereum“ atvirojo kodo biblioteką ir „Infura“ paslaugų tiekėją, vykdo transakcijų stebėseną ir inicijuoja transakcijų pakeitimą mokesčiu mechanizmą.
5. Atlikus pasiūlyto metodo vertinimą, nustatyta, kad decentralizuotų sandorių apsaugos sistema efektyviausiai veikia su transakcijų pagal požymius nustatymo metodu, kuris eksperimento metu pasirodė pranašesnis prieš išankstinių sandorių simuliacijos metodą tiek tikslumo (78,6% prieš 60%), tiek jautrumo (91,7% prieš 50%) apsektais. Esminis pasiūlyto metodo privalumas yra koncepcijos įrodymas, jog transakcijų pakeitimo mokesčiu metodas gali būti pritaikomas gynybos mechanizmų kūrime (75% sėkmės rodiklis), bet identifikuotos tikėtinos techninės tinklo pralaidumo ir vėlavimo bei eksperimento metu apibrėžtos griežtos sąlygos sutrikdė pasiekti teorinį efektyvumo lūkestį.

PASIŪLYMAI

1. Nors eksperimentinis tyrimas patvirtina sistemos veiksmingumą gintis prieš automatizuotos prekybos sistemas, tačiau neįvertintas ekonominis naudingumas. Tai reiškia, kad nors ir sistema gali padėti apsiginti nuo atakų, tai nereiškia, kad sprendimas yra efektyvus sistemos naudotojui ekonomine prasme gintis naudojantis sistema, nes gynybos kaštai gali neatpirkti žalos dydžio ir papildomų transakcijų mokesčių, bandant įvykdyti pakartotinius žetonų keitimo sandorius, tad prieš taikant sprendimą reiktų įvertinti šį ekonominės naudos aspektą.
2. Siekiant padidinti sistemos efektyvumą nustatant teisingas atakas, svarstytinas kombinuotas atakų atpažinimo metodas, kuris apjungtų transakcijų atpažinimo pagal požymius metodą su išankstine sandorių simuliacija, siekiant nustatyti kainos pokytį decentralizuotoje biržoje.

LITERATŪRA

1. Alharby, Maher ir van Moorsel, Aad. 2017. Blockchain-based Smart Contracts: A Systematic Mapping Study. 2017.
2. Auditing with Smart Contracts. Rozario, Andrea M. ir Vasarhelyi, Miklos A. 2018. s.l. : The International Journal of Digital Accounting Research, 2018 m.
3. Bartoletti, Massimo ir Zunino, Roberto. 2023. A theoretical basis for MEV. 2023.
4. Blockchain-based database to ensure data integrity in cloud computing environments. Gaetani, Edoardo, et al. 2017. s.l. : Italian Conference on Cybersecurity, 2017.
5. Capponi, Agostino, Jia, Ruizhe ir Wang, Ye. 2022. The Evolution of Blockchain: from Lit to Dark. 2022.
6. Christidis, Konstantinos ir Devetsikiotis, Michael. 2019. Blockchains and Smart Contracts for the Internet of Things. 2019.
7. Daian, Philip, et al. 2019. Flash Boys 2.0: Frontrunning, Transaction Reordering, and Consensus Instability in Decentralized Exchanges. 2019.
8. Design for Trust: An Exploration of the Challenges and Opportunities of Bitcoin Users. Sas, Corina ir Khairuddin, Irni Eliana. 2017. s.l. : ACM Conference Computer Human Interaction 2017, 2017.
9. Eberhardt, Jacob ir Tai, Stefan. 2017. On or Off the Blockchain? Insights on Off-Chaining Computation and Data. 2017.
10. Europos Parlamento ir Tarybos Reglamentas (ES) 2023/1114 dėl kriptoturto rinkų. Europos Parlamentas ir Taryba. 2023. Europos Sąjungos oficialusis leidinys : s.n., 2023 m., Europos Sąjungos oficialusis leidinys, p. 40-205.
11. Exploring Ethereum's Data Stores: A Cost and Performance Comparison. Kostamis, Periklis, Sendros, Andreas ir Efraimidis, Pavlos S. 2021. s.l. : 3rd Conference on Blockchain Research & Applications for Innovative Networks and Services, 2021.
12. Gao, Lixia. 2025. Enterprise internal audit data encryption based on blockchain technology. 2025.
13. Gomber, Peter, et al. 2011. High-Frequency Trading. 2011.
14. Gomez-Trujillo, Ana Maria, Velez-Ocampo, Juan ir Alejandra, Maria. 2021. Trust, transparency, and technology: blockchain and its relevance in the context of the 2030 agenda. s.l. : The Palgrave Handbook of Corporate Sustainability in the Digital Era, 2021.
15. Yang, Sen, et al. 2022. SoK: MEV Countermeasures: Theory and Practice. 2022.

16. Jones, Charles M. 2013. What Do We Know About High-Frequency Trading? 2013.
17. Making Smart Contracts Smarter. Luu, Loi, et al. 2016. s.l. : Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016.
18. Nakamoto, Satoshi. Bitcoin: A peer-to-peer electronic cash system. 2008 : s.n.
19. Narayanan, Arvind, et al. 2016. Bitcoin and Cryptocurrency Technologies. 2016.
20. Parizi, Reza M., Singh, Amritraj ir Dehghantanha, Ali. 2018. Smart Contract Programming Languages on Blockchains: An Empirical Evaluation of Usability and Security. 2018.
21. Park, Seongwan, et al. 2023. Unraveling the MEV Enigma: ABI-Free Detection Model using Graph Neural Networks. 2023.
22. Powers, David M. W. 2020. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. 2020 m.
23. Qin, Kaihua, Zhou, Liyi ir Gervais, Arthur. 2021. Quantifying Blockchain Extractable Value: How dark is the forest? 2021.
24. Revealing and Concealing Bitcoin Identities: A Survey of Techniques. Bergman, Karolina ir Rajput, Saeed. 2021. s.l. : Proceedings of the 3rd ACM International Symposium on Blockchain and Secure Critical Infrastructure, 2021.
25. Salehi, Mehdi, Clark, Jeremy ir Mannan, Mohammad. 2022. Not so immutable: Upgradeability of Smart Contracts on Ethereum. 2022.
26. Shayan, Eskandari, Moosavi, Seyedehmahsa ir Clark, Jeremy. 2019. SoK: Transparent Dishonesty: front-running attacks on Blockchain. 2019.
27. Taherdoost, Hamed. 2023. Smart Contracts in Blockchain Technology: A Critical Review. 2023.
28. Torres, Christof Ferreira, Camino, Ramiro ir State, Radu. 2021. Frontrunner Jones and the Raiders of the Dark Forest: An Empirical Study of Frontrunning on the Ethereum Blockchain. 2021.
29. Weintraub, Ben, et al. 2022. A flash(bot) in the pan: measuring maximal extractable value in private pools. IMC '22: Proceedings of the 22nd ACM Internet Measurement Conference. 2022 m.
30. Werner, Sam M., et al. 2021. SoK: Decentralized Finance (DeFi). 2021.
31. Xu, Xiwei, et al. 2016. The Blockchain as a Software Connector. 2016.
32. Zhang, Mengqian, et al. 2025. s.l. : The Twenty-Sixth ACM Conference on Economics and Computation, 2025.

33. Zheng, Zibin, et al. 2017. An overview of blockchain technology: Architecture, consensus, and future trends. s.l. : IEEE international congress on big data (BigData congress), 2017.
34. Zhou, Liyi, et al. 2020. High-Frequency Trading on Decentralized On-Chain Exchanges. 2020 m.

PRIEDAI

1 PRIEDAS

PROGRAMINIS APSAUGOS SISTEMOS IŠMANIOJO KONTRAKTO KODAS

```
pragma solidity >=0.8.30 <0.9.0;

interface IERC20 {
    function transferFrom(address sender, address recipient, uint256 amount) external returns (bool);
    function transfer(address recipient, uint256 amount) external returns (bool);
    function approve(address spender, uint amount) external returns (bool);
}

interface IUniswapV2Router02 {
    function swapExactTokensForTokens(uint amountIn, uint amountOutMin, address[] calldata path,
address to, uint deadline) external returns (uint[] memory amounts);
}

contract Guard {
    address public ownerAddress;

    uint256 public previousOrderId = 0;
    mapping(uint256 => Order) public orders;

    constructor() {
        ownerAddress = msg.sender;
    }

    modifier onlyOwner() {
        require(msg.sender == ownerAddress, "Metodas skirtas tik savininkui");
        _;
    }
```

```
struct Order {
        uint256 orderId;
        address sender;
        address router;
        address tokenIn;
        address tokenOut;
        uint256 amountIn;
        uint256 minAmountOut;
        uint256 deadline;
        bool finished;
    }

    event OrderPlaced(Order order);

    function placeOrder(
        address router,
        address tokenIn,
        address tokenOut,
        uint256 amountIn,
        uint256 minAmountOut,
        uint256 deadline
    ) external {
        require(amountIn > 0, "Suma turi buti teigiama");

        bool transferSuccessful = IERC20(tokenIn).transferFrom(msg.sender, address(this), amountIn);
        require(transferSuccessful, "Nepavyko perimti lesu");

        uint256 orderId = ++previousOrderId;
        Order memory newOrder = Order({
            orderId: orderId,
            sender: msg.sender,
```

```
        router: router,
        tokenIn: tokenIn,
        tokenOut: tokenOut,
        amountIn: amountIn,
        minAmountOut: minAmountOut,
        deadline: deadline,
        finished: false
    });
    orders[orderId] = newOrder;

    emit OrderPlaced(newOrder);
}

function retrieveMoney() external {
    for (uint256 i = 1; i <= previousOrderId; i++) {
        Order storage order = orders[i];

        if (order.sender == msg.sender && order.finished == false) {
            order.finished = true;

            IERC20(order.tokenIn).transfer(msg.sender, order.amountIn);
        }
    }
}

function executeOrder(uint256 orderId) external onlyOwner {
    Order storage order = orders[orderId];
    require(order.finished == false, "Order nebegalioja");
    order.finished = true;

    address[] memory path = new address[](2);
```

```
        path[0] = order.tokenIn;
        path[1] = order.tokenOut;

        IERC20(order.tokenIn).approve(address(order.router), order.amountIn);
        IUniswapV2Router02(order.router).swapExactTokensForTokens(
            order.amountIn,
            order.minAmountOut,
            path,
            order.sender,
            order.deadline
        );
    }

    function cancelOrder(uint256 orderId) external onlyOwner {
        Order storage order = orders[orderId];
        require(order.finished == false, "Order nebegalioja");
        order.finished = true;
        IERC20(order.tokenIn).transfer(msg.sender, order.amountIn);
    }
}
```

IŠORINIO SISTEMOS VALDIKLIO PROGRAMINIS KODAS

Programos įėjimo taškas *Program.cs*

```
using Fintech.Console.Events;
using Fintech.Console.Mempool;
using Nethereum.Web3;
using Nethereum.Web3.Accounts;

namespace Fintech.Console;

class Program
{
    private const string _infuraKey = "";
    private const string _contractAddress = "";
    private const string _victimPrivateKey = "";

    static async Task Main(string[] args)
    {
        var httpUrl = $"https://sepolia.infura.io/v3/{_infuraKey}";
        var wssUrl = $"wss://sepolia.infura.io/ws/v3/{_infuraKey}";
        var eventWeb3 = new Web3(httpUrl);
        var victimAccount = new Account(_victimPrivateKey);
        var victimWeb3 = new Web3(victimAccount, httpUrl);

        await MempoolScreener.StartMempoolScreenerAsync(eventWeb3, wssUrl);
        await EventConsumer
            .ConsumeEventsAsync<OrderPlacedEvent>(
                eventWeb3,
                _contractAddress,
                (e) =>
                {
```

```
                _ = TransactionExecutor.ExecuteOrderAsync(
                    eventWeb3,
                    victimWeb3,
                    victimAccount,
                    _contractAddress,
                    e);
            },
            CancellationToken.None);
    }
}
```

Atakų atpažinimo modulis (pagal transakcijų požymius) ***ParameterDetection.cs***

```
using Fintech.Console.Models;

namespace Fintech.Console.Detections;

public static class ParameterDetection
{
    private const int _threshold = 10;

    public static Task<bool> EvaluateTransactionsAsync(
        SwapTransaction originalTransaction,
        List<SwapTransaction> interimTransactions)
    {
        var isAttack = interimTransactions
            .Any(t =>
                t.RouterAddress == originalTransaction.RouterAddress
                && t.TransactionHash != originalTransaction.TransactionHash
                && t.TokenIn == originalTransaction.TokenIn
                && t.TokenOut == originalTransaction.TokenOut
                && t.GasPrice.Value > (originalTransaction.GasPrice.Value * (100 + _threshold)/ 100));
```

```
        return Task.FromResult(isAttack);
    }
}
```

Atakų atpažinimo modulis (išankstinių sandorių simuliacija) ***SimulationDetection.cs***

```
using System.Numerics;
using Fintech.Console.Functions;
using Fintech.Console.Models;
using Nethereum.Web3;

namespace Fintech.Console.Detections;

public class SimulationDetection
{
    private const int _threshold = 3;
    private const string _tradableToken = "";
    private const string _pairAddress = "";

    public static async Task<bool> EvaluateTransactionsAsync(
        Web3 web3,
        List<SwapTransaction> interimTransactions)
    {
        var reserves = await GetPoolSizeAsync(web3, _pairAddress);

        return interimTransactions
            .Any(it =>
                AnalyzeTransaction(reserves, it));
    }
```

```
    private static bool AnalyzeTransaction(
        (BigInteger Reserve0, BigInteger Reserve1) reserves,
        SwapTransaction transaction)
    {
        var otherToken = transaction.TokenIn.Equals(_tradableToken,
StringComparison.OrdinalIgnoreCase)
            ? transaction.TokenOut
            : transaction.TokenIn;

        var isTradableToken0 = string.Compare(_tradableToken, otherToken,
StringComparison.OrdinalIgnoreCase) < 0;

        BigInteger inputReserve;

        if (transaction.TokenIn.Equals(_tradableToken, StringComparison.OrdinalIgnoreCase))
        {
            inputReserve = isTradableToken0 ? reserves.Reserve0 : reserves.Reserve1;
        }
        else
        {
            inputReserve = isTradableToken0 ? reserves.Reserve1 : reserves.Reserve0;
        }

        var amountInDec = (decimal)transaction.Amount;
        var reserveInDec = (decimal)inputReserve;

        if (reserveInDec == 0)
        {
            return false;
        }
```

```
        var priceImpact = (amountInDec / reserveInDec) * 100M;

        return priceImpact >= _threshold;
    }

    private static async Task<(BigInteger Reserve0, BigInteger Reserve1)> GetPoolSizeAsync(
        Web3 web3,
        string pairAddress)
    {
        var getReservesMessage = new GetReservesFunction();
        var queryHandler = web3.Eth.GetContractQueryHandler<GetReservesFunction>();
        var reserves = await queryHandler.QueryDeserializingToObjectAsync<GetReservesOutput>(
            getReservesMessage,
            pairAddress
        );

        return (reserves.Reserve0, reserves.Reserve1);
    }
}
```

Tinklo stebėjimo modulis *MempoolScreener.cs*

```
using Fintech.Console.Functions;
using Fintech.Console.Models;
using Nethereum.Contracts;
using Nethereum.JsonRpc.Client.Streaming;
using Nethereum.JsonRpc.WebSocketStreamingClient;
using Nethereum.RPC.Eth.Subscriptions;
using Nethereum.Web3;

namespace Fintech.Console.Mempool;
```

```
public static class MempoolScreener
{
    private static List<(DateTimeOffset DateTime, SwapTransaction Transaction)> _swapTransactions
= new();
    private static object _lock = new object();
    private static StreamingWebSocketClient _client;

    private const int _minutesToHoldData = 1;

    public static async Task StartMempoolScreenerAsync(
        Web3 web3, string wssUrl)
    {
        _client = new StreamingWebSocketClient(wssUrl);
        var subscription = new EthNewPendingTransactionSubscription(_client);

        subscription.SubscriptionDataResponse += async (object _, StreamingEventArgs<string> e) =>
        {
            var txHash = e.Response;
            var tx = await web3.Eth.Transactions.GetTransactionByHash.SendRequestAsync(txHash);

            if (tx is { To: not null })
            {
                var decoder = new SwapExactTokensForTokensFunction();
                var data = decoder.DecodeInput(tx.Input);
                var swapTransaction = new SwapTransaction
                {
                    TransactionHash = tx.TransactionHash,
                    RouterAddress = tx.To,
                    Amount = data.AmountIn,
                    TokenIn = data.Path.First(),
                    TokenOut = data.Path.Last(),
```

```
                GasPrice = tx.GasPrice
            };

            lock (_lock)
            {
                var date = DateTimeOffset.UtcNow;
                _swapTransactions.Add((date, swapTransaction));
                _swapTransactions.RemoveAll(st => st.DateTime.AddMinutes(_minutesToHoldData) <
date);
            }
        }
    };

    await _client.StartAsync();
    await subscription.SubscribeAsync();
  }

  public static List<(DateTimeOffset DateTime, SwapTransaction Transaction)>
GetSwapTransactions()
  {
    lock (_lock)
    {
      var date = DateTimeOffset.UtcNow;
      _swapTransactions.RemoveAll(st => st.DateTime.AddMinutes(_minutesToHoldData) <
date);

      return _swapTransactions.ToList();
    }
  }
}
```

Transakcijų vykdymo modulis TransactionExecutor.cs

```
using System.Numerics;
using Fintech.Console.Detections;
using Fintech.Console.Events;
using Fintech.Console.Functions;
using Fintech.Console.Mempool;
using Fintech.Console.Models;
using Nethereum.Hex.HexTypes;
using Nethereum.RPC.Eth.DTOs;
using Nethereum.Util;
using Nethereum.Web3;
using Nethereum.Web3.Accounts;

namespace Fintech.Console;

public class TransactionExecutor
{
    private const int _milisecondScan = 4000;

    public static async Task ExecuteOrderAsync(
        Web3 eventWeb3,
        Web3 victimWeb3,
        Account victimAccount,
        string contractAddress,
        OrderPlacedEvent orderPlacedEvent)
    {
        var nonce = await victimWeb3.Eth.Transactions.GetTransactionCount.SendRequestAsync(
            victimAccount.Address,
            BlockParameter.CreatePending()
        );
        var currentGasPrice = await victimWeb3.Eth.GasPrice.SendRequestAsync();
```

```
        var executeOrderHandler =
victimWeb3.Eth.GetContractTransactionHandler<ExecuteOrderFunction>();
        var executeOrderFunction = new ExecuteOrderFunction { OrderId =
orderPlacedEvent.Order.OrderId };
        var executeOrderTransaction = await
executeOrderHandler.CreateTransactionInputEstimatingGasAsync(contractAddress,
executeOrderFunction);
        executeOrderTransaction.Nonce = nonce;
        executeOrderTransaction.GasPrice = currentGasPrice;
        var executeOrderTransactionHash = await
victimWeb3.TransactionManager.SendTransactionAsync(executeOrderTransaction);

        var originalTransaction = new SwapTransaction
        {
            TransactionHash = executeOrderTransactionHash,
            RouterAddress = orderPlacedEvent.Order.Router,
            Amount = orderPlacedEvent.Order.AmountIn,
            TokenIn = orderPlacedEvent.Order.TokenIn,
            TokenOut = orderPlacedEvent.Order.TokenOut,
            GasPrice = currentGasPrice
        };

        var interimTransactions = await GetInterimTransactionsAsync(originalTransaction,
_milisecondScan);
        //var isAttacked = await ParameterDetection.EvaluateTransactionsAsync(originalTransaction,
interimTransactions);
        var isAttacked = await SimulationDetection.EvaluateTransactionsAsync(eventWeb3,
interimTransactions);

        if (!isAttacked)
```

```
        {
            return;
        }

        var cancelOrderHandler =
victimWeb3.Eth.GetContractTransactionHandler<CancelOrderFunction>();
        var cancelOrderFunction = new CancelOrderFunction { OrderId =
orderPlacedEvent.Order.OrderId };
        var cancelOrderTransaction = await
cancelOrderHandler.CreateTransactionInputEstimatingGasAsync(contractAddress,
cancelOrderFunction);
        cancelOrderTransaction.Nonce = nonce;
        cancelOrderTransaction.GasPrice = new HexBigInteger((currentGasPrice.Value * 125) / 100);
        await victimWeb3.TransactionManager.SendTransactionAsync(cancelOrderTransaction);
    }

    private static async Task<List<SwapTransaction>> GetInterimTransactionsAsync(
        SwapTransaction originalTransaction,
        int milisecondLimit)
    {
        await Task.Delay(milisecondLimit);

        var dateFrom = DateTime.UtcNow.AddMilliseconds(-milisecondLimit);

        return MempoolScreener
            .GetSwapTransactions()
            .Where(st =>
                st.DateTime >= dateFrom
                && st.Transaction.TransactionHash != originalTransaction.TransactionHash)
            .Select(st => st.Transaction)
            .ToList();
```

```
    }
}
```

Išmanaus kontrakto įvykių stebėjimo modulis *EventConsumer.cs*

```
using Nethereum.ABI.FunctionEncoding.Attributes;
using Nethereum.Hex.HexTypes;
using Nethereum.RPC.Eth.DTOs;
using Nethereum.Web3;

namespace Fintech.Console.Events;

public static class EventConsumer
{
    public static async Task ConsumeEventsAsync<T>(
        Web3 web3,
        string contractAddress,
        Action<T> handler,
        CancellationToken cancellationToken)
        where T : IEventDTO, new()
    {
        var currentBlockNumber = await web3.Eth.Blocks.GetBlockNumber.SendRequestAsync();
        var lastCheckedBlock = currentBlockNumber.Value;
        var eventHandler = web3.Eth.GetEvent<T>(contractAddress);

        while (!cancellationToken.IsCancellationRequested)
        {
            var latestBlockNumberHex = await
web3.Eth.Blocks.GetBlockNumber.SendRequestAsync();
            var latestBlock = latestBlockNumberHex.Value;
```

```
        if (latestBlock > lastCheckedBlock)
        {
            var fromBlock = new BlockParameter(new HexBigInteger(lastCheckedBlock + 1));
            var toBlock = new BlockParameter(latestBlockNumberHex);
            var filterInput = eventHandler.CreateFilterInput(fromBlock, toBlock);
            var logs = await eventHandler.GetAllChangesAsync(filterInput);

            if (logs.Count > 0)
            {
                foreach (var log in logs)
                {
                    handler(log.Event);
                }
            }

            lastCheckedBlock = latestBlock;
        }

        await Task.Delay(2000, cancellationToken);
    }
  }
}
```

Išmaniųjų sutarčių funkcijų aprašų rinkinys *Functions.cs*

```
using System.Numerics;
using Nethereum.ABI.FunctionEncoding.Attributes;
using Nethereum.Contracts;

namespace Fintech.Console.Functions;

[Function("cancelOrder")]
```

```
public class CancelOrderFunction : FunctionMessage
{
    [Parameter("uint256", "orderId", 1)]
    public BigInteger OrderId { get; set; }
}

[Function("executeOrder")]
public class ExecuteOrderFunction : FunctionMessage
{
    [Parameter("uint256", "orderId", 1)]
    public BigInteger OrderId { get; set; }
}

[Function("getReserves", typeof(GetReservesOutput))]
public class GetReservesFunction : FunctionMessage
{
}

[FunctionOutput]
public class GetReservesOutput : IFunctionOutputDTO
{
    [Parameter("uint112", "_reserve0", 1)]
    public BigInteger Reserve0 { get; set; }

    [Parameter("uint112", "_reserve1", 2)]
    public BigInteger Reserve1 { get; set; }

    [Parameter("uint32", "_blockTimestampLast", 3)]
    public uint BlockTimestampLast { get; set; }
}
```

```
[Function("swapExactTokensForTokens", "uint256[]")]
public class SwapExactTokensForTokensFunction : FunctionMessage
{
    [Parameter("uint256", "amountIn", 1)]
    public BigInteger AmountIn { get; set; }

    [Parameter("uint256", "amountOutMin", 2)]
    public BigInteger AmountOutMin { get; set; }

    [Parameter("address[]", "path", 3)]
    public string[] Path { get; set; }

    [Parameter("address", "to", 4)]
    public string To { get; set; }

    [Parameter("uint256", "deadline", 5)]
    public BigInteger Deadline { get; set; }
}
```

Išmaniojo kontrakto įvykio aprašas OrderPlacedEvent.cs

```
using System.Numerics;
using Nethereum.ABI.FunctionEncoding.Attributes;

namespace Fintech.Console.Events;

[Event("OrderPlaced")]
public class OrderPlacedEvent: IEventDTO
{
    [Parameter("tuple", "order", 1, false)]
    public OrderStruct Order { get; set; }
}
```

```
public class OrderStruct
{
    [Parameter("uint256", "orderId", 1)]
    public BigInteger OrderId { get; set; }

    [Parameter("address", "sender", 2)]
    public string Sender { get; set; }

    [Parameter("address", "router", 3)]
    public string Router { get; set; }

    [Parameter("address", "tokenIn", 4)]
    public string TokenIn { get; set; }

    [Parameter("address", "tokenOut", 5)]
    public string TokenOut { get; set; }

    [Parameter("uint256", "amountIn", 6)]
    public BigInteger AmountIn { get; set; }

    [Parameter("uint256", "minAmountOut", 7)]
    public BigInteger MinAmountOut { get; set; }

    [Parameter("uint256", "deadline", 8)]
    public BigInteger Deadline { get; set; }

    [Parameter("bool", "finished", 9)]
    public bool Finished { get; set; }
}
```

Duomenų perdavimo objektas SwapTransaction.cs

```
using System.Numerics;
using Nethereum.Hex.HexTypes;

namespace Fintech.Console.Models;

public class SwapTransaction
{
    public string TransactionHash { get; set; }

    public string RouterAddress { get; set; }

    public BigInteger Amount { get; set; }

    public string TokenIn { get; set; }

    public string TokenOut { get; set; }

    public HexBigInteger GasPrice { get; set; }
}
```

PROGRAMINIS KURIAMO ERC20 ŽETONO IŠMANIOJO KONTRAKTO KODAS

```
pragma solidity >=0.8.30 <0.9.0;


import "@openzeppelin/contracts/token/ERC20/ERC20.sol";


contract MasterThesis is ERC20 {
    constructor() ERC20("Magistras2025", "MBD25") {
        _mint(msg.sender, 1000000000 * (10 ** uint256(decimals())));
    }
}
```

EKSPERIMENTE MODELIUOTŲ SCENARIJŲ PARAMETRAI

Scenarijaus nr.	Scenarijaus tipas	Keičiamas žetonas	Kainos koeficientas	Kiekio koeficientas
1	Ataka	Identiškas	1,5	1
2	Ataka	Identiškas	2	2
3	Ataka	Identiškas	1,15	1
4	Ataka	Identiškas	1,3	5
5	Ataka	Identiškas	1,4	5
6	Ataka	Identiškas	1,2	10
7	Ataka	Identiškas	3	1
8	Ataka	Identiškas	1,25	2
9	Ataka	Identiškas	1,18	1
10	Ataka	Identiškas	1,22	1,5
11	Ataka	Identiškas	1,35	3
12	Ataka	Identiškas	1,12	1
13	Ribinė	Nesutampantis	1	1
14	Ribinė	Nesutampantis	6	0,5
15	Ribinė	Identiškas	0,9	1
16	Ribinė	Identiškas	0,5	0,1
17	Ribinė	Nesutampantis	1,1	5
18	Ribinė	Identiškas	1,5	0,01
19	Įprasta	Identiškas	1	1
20	Įprasta	Nesutampantis	1,2	0,5
21	Įprasta	Identiškas	0,95	2
22	Įprasta	Nesutampantis	2	10
23	Įprasta	Identiškas	1,02	0,5
24	Įprasta	Nesutampantis	1,1	1

Scenarijaus nr.	**Scenarijaus tipas**	**Keičiamas žetonas**	**Kainos koeficientas**	**Kiekio koeficientas**
25	Įprasta	Identiškas	1,5	0,1
26	Įprasta	Identiškas	1,01	5
27	Įprasta	Identiškas	1,02	10
28	Įprasta	Identiškas	1,5	2
29	Įprasta	Identiškas	1,001	1
30	Įprasta	Identiškas	2	0,001

ATAKŲ NUSTATYMO PAGAL TRANSAKCIJŲ POŽYMIUS METODO TESTAVIMO TRANSAKCIJOS

Pateikiamos testavimo metu įvykusių transakcijų maišos ir, jei vyko atšaukimo transakcija, sėkmės rezultatas. Nepateikta atšaukimo transakcijos maiša scenarijuose indikuoja, jog ataka nebuvo nustatyta.

Scenarijus nr. 1

- Inicijavimo transakcijos maiša: 0x7767aa82066489c61f9984a6ff53171c689236530be0f4e94ab841622cce1349
- Keitimo transakcijos maiša: 0x4baac5608c12323cc0ec3ddac41fa221a2f404f1eed7aff339fa7e5496c89c4b
- Atakos transakcijos maiša: 0x5606cbb13e232437e8ca6cdaab0ede2f8fa9ced9a5b3812d22e4b1ff80ec0e4a
- Atšaukimo transakcijos maiša (sėkminga): 0x265c02f6ae0ced17e56642022de39b231f8c84312122ae6a4c66aedfa582d583

Scenarijus nr. 2

- Inicijavimo transakcijos maiša: 0x36463151bd600253cef15a245643c6c81d481271fa4395284f5ef60882d0161b
- Keitimo transakcijos maiša: 0x47d0c806f0f25e4eedb1d8ce78920c0ba05cf86718bd76ddad3690ce1e727f16
- Atakos transakcijos maiša: 0xba4522e179ef1de912d3b1f6c298b7cb3fed3d132d6c88a7461c76d19576cb7f
- Atšaukimo transakcijos maiša (nepavykusi): 0x71b2b8967fee14bab37af3a6302219e81158692e705c7da1432689c47e33bee4

Scenarijus nr. 3

- Inicijavimo transakcijos maiša:
  0x76fd31631edc44361159987a5482bff68c8f959a91b79776ee3ac53144cbd763
- Keitimo transakcijos maiša:
  0xec24f2eefe5ddc91ce3e1aaa4f12e27b3a178b97adcebb8569cc9a7f36d7644a
- Atakos transakcijos maiša:
  0x557cd7546a6199209b35d288d5d7ef0f79bae5347678a698aeaa17867d700d7c
- Atšaukimo transakcijos maiša (sėkminga):
  0xd3083670b157f440725bdafc0c813cc398a95781a28889a165c4bf053e30f6ae

Scenarijus nr. 4

- Inicijavimo transakcijos maiša:
  0x8094747580aaf7acda89083077dccc21cfb127bbc61bffe5cbc7c8b735a7b3c6
- Keitimo transakcijos maiša:
  0x368757bc379deafdd2a164a6bdec0b0d992b3fa45a343a5cd4617eb63c2dee05
- Atakos transakcijos maiša:
  0x5daed972e0223ee8f83591d0b02bc74628ef6a451efabbb1f595f3b2d17893c9
- Atšaukimo transakcijos maiša (sėkminga):
  0x1c60250def6eecd9c19cc70b4c885b6f9c54eb3a3ae037ae0dc37fc05d98e785

Scenarijus nr. 5

- Inicijavimo transakcijos maiša:
  0xd29abeff838577e18f364319b75fc06da8d30569b9617cce9737a740f0675c30
- Keitimo transakcijos maiša:
  0x900a3273430738008eda42776421e7f2b999d47edb1fbcea5770273f7636dab7
- Atakos transakcijos maiša:
  0x8368cc887c13ce9f72a6f5aee97cb0c35c1961ed22707ecb3e27ec6f35a7d8c1

Scenarijus nr. 6

- Inicijavimo transakcijos maiša:
  0x2fb9e561622dec0397530ffed63bd6ec3827228ce7d2801b732ca1c6bb3941d8
- Keitimo transakcijos maiša:
  0xa13b9d8011552ec723914760632104eacc87dd26fcc244bea931c77672925b27
- Atakos transakcijos maiša:
  0x3540b0fd3dfca8ffab6475621d06baaaf07d6cb634a41b28ff939882896d3398
- Atšaukimo transakcijos maiša (sėkminga):
  0x555ad95a799b53f4a0dd4ceede6d3d8dbf1a9eb1e0b7931d84c8a3eee9f19d40

Scenarijus nr. 7

- Inicijavimo transakcijos maiša:
  0x42c0b9f30433a6d8788118bd956bcbe2cc143e81422fae30b3b82bfe39b23c42
- Keitimo transakcijos maiša:
  0xaf5578ca5009c77f9c522d719726be11a1110a9d0a96e78d151f1264cfb3eb04
- Atakos transakcijos maiša:
  0x97b596067782de52f812ebbe831c2c6437bb6742b394cb3dcc0002b632195ab7
- Atšaukimo transakcijos maiša (sėkminga):
  0x51c43c9537e718f6050d1b0f81f9ed860ce827398da415b21e8f68e34cbf5436

Scenarijus nr. 8

- Inicijavimo transakcijos maiša:
  0x2d2db4208fbd9bc773458e9e659d84bad95fbc321c8c49b02f2223663878ea60
- Keitimo transakcijos maiša:
  0xe79a40438a9e09a24f00a0000550f2bfa133fbd379b6e6422bf156bd4f75a18b
- Atakos transakcijos maiša:
  0x25b8024d9ccecbb026712f8613c14e9f0302eb3200ae2472c5859af022144b3d
- Atšaukimo transakcijos maiša (nepavykusi):
  0x6974da53f27781c9ec7ecf537e02cbfd0881cbbe2ba153bfc9fa6d5d0f8c3f11

Scenarijus nr. 9

- Inicijavimo transakcijos maiša:
  0xc5e8c946af7f50a53ccb60017b3bbb69ad0fcbb295ccf4bf6c1171e3f0f96ba6
- Keitimo transakcijos maiša:
  0xa5ece83fd6a753abc5863dd9d3c281a99bae428b7a384512a23f076595758ee5
- Atakos transakcijos maiša:
  0x6f0db17363657e14df8602d63ad995afd2b60a8a319d296aa80b62697f4b5c51
- Atšaukimo transakcijos maiša (nepavykusi):
  0xc5cc3fc523b27ec4d1996c2befb5f693374f0469a98152584dd3f52a5acafec6

Scenarijus nr. 10

- Inicijavimo transakcijos maiša:
  0x0090be1923877d79498ef7c1b95db69eba0ce853bbca9548611e27047241f194
- Keitimo transakcijos maiša:
  0x2502da8cd4a4020c3c8b867f4b541a803c6ed157b45748c4692df931e6583aa7
- Atakos transakcijos maiša:
  0xb4af843de570b25f520d7324c5abe0ff1c92e00bf67521adaed17219f85499eb
- Atšaukimo transakcijos maiša (sėkminga):
  0x9c51c637c05d929d4e6ecc815f739fca1a6e129030124d37b12e363b075752a2

Scenarijus nr. 11

- Inicijavimo transakcijos maiša:
  0xd75a7e5d3630c11557c8aded817de9af291a25a1dbe9f47c3229f1fbfa907d05
- Keitimo transakcijos maiša:
  0x7acaeb0a58b3c44967f4b9af4932873cfdde5caf29f2a1ce36acff865c6fc5bb
- Atakos transakcijos maiša:
  0xdc2c1a635c19ee0450f290aa72e591f95aed2e84c87cfb5f0f4e5e302a41e903
- Atšaukimo transakcijos maiša (sėkminga):
  0x422cccd404d825125c77a1a462814725dacaa9b394209d680a02342c178aa81a

Scenarijus nr. 12

- Inicijavimo transakcijos maiša:
  0x6aa70e88fddcbf81da672419c9e302e07f2e2971b44a42072a71d5ccd4bdf296
- Keitimo transakcijos maiša:
  0xa7265c417fe03a7feceafb8e9b41cf965d619595cea96da861f1c9ed415488ee
- Atakos transakcijos maiša:
  0x6e567be0f94bd9972e8005095336fcab722af09bc9005ba18c6bd905441bc357
- Atšaukimo transakcijos maiša (sėkminga nepavykusi):
  0x8f8bde0e75764597c3e3805dc2519c4fd2dfc27300aed3f051e956a377841398

Scenarijus nr. 13

- Inicijavimo transakcijos maiša:
  0x0e4759e68205401258l6be627401875c1e13147471bb27e4f458423911f828ea
- Keitimo transakcijos maiša:
  0xf581b717e8cb0e1b10e9ec1bcf19e4f34c337506e3f77851b0c826ef38eb418e
- Atakos transakcijos maiša:
  0xa62611e67c0f3b8a0e4498260a0cbd802d54b45c00705dded1636b8d434a3a1f

Scenarijus nr. 14

- Inicijavimo transakcijos maiša:
  0x6dc83fd466cfa7ff55cab3071169cadca19e740455eecfeeba20502e4f083d12
- Keitimo transakcijos maiša:
  0x532b71f7bf2af4d90a1f6ae0c03c11a8e596b0d3e18519a794a72a782f28380c
- Atakos transakcijos maiša:
  0xc78d9062c44288bc83dda0e4ca7b62473f61e5998610fb3c9133b4b8d0bc19fc

Scenarijus nr. 15

- Inicijavimo transakcijos maiša:
  0x1ab1e34febf40634c10f5d55458815c0896e10454c15c937db1984c71c468848
- Keitimo transakcijos maiša:
  0x9c6d2b2dbb1d51d96b8acb0d05b0929f25a80bb683a2972fa110036f832f03a0
- Atakos transakcijos maiša:
  0x1232e3982c025e2d332c3dc08e1cf4825b3932632b05f5b7012afc895603e6ab

Scenarijus nr. 16

- Inicijavimo transakcijos maiša: 0x2f3374aebad90676bf4d8f81c5d9960a406f3867fa85b9fa845553d28348d0a5
- Keitimo transakcijos maiša: 0xbe164a67b9ceccdac7dbe6afc7509810ee02b97f4e7e3219321812fcebcf2585
- Atakos transakcijos maiša: 0xe815553179c33887c3129aafa95256045ddb39853aa0bb0c7998b370b19d442d

Scenarijus nr. 17

- Inicijavimo transakcijos maiša: 0xef0f0bcd104108b8a1e54ac8f51e580519d7eb457f2dfcf590456b0333e663e2
- Keitimo transakcijos maiša: 0x8881428a44c949484d477e2ca3a3d07434e7e23df14faf921572eaa35707b6c4
- Atakos transakcijos maiša: 0xa5b3fafb2f8eb0d78bd5282016c4053d19addb09cf6a0d664169694845f57110

Scenarijus nr. 18

- Inicijavimo transakcijos maiša: 0x12efb05b82f9a42cc42e3a31d51b193a3236f79de591c517c6c088127be2d20f
- Keitimo transakcijos maiša: 0x56fa24321b6b884dabe1852f42fd950142f2619351fcf7844b23edc79d8b6338
- Atakos transakcijos maiša: 0x44b0849424e59e05fdb02c3cb296a025e3387bb3395d49bcda0318456895aafa
- Atšaukimo transakcijos maiša (sėkminga): 0x902c8b629acbe3ae11acacd918ae1e9486daaa0f37a10367eb41f84590c9cd1c

Scenarijus nr. 19

- Inicijavimo transakcijos maiša: 0xf2722d3129d8f378133f1f95925bcb73e30b2f5525ee240e10829c4fa200fedd
- Keitimo transakcijos maiša: 0x772b97b0aa5150fad5638c4840b420b87328c970805a5f4035c5341738d74ee9
- Atakos transakcijos maiša: 0x6672a47401727373dd6673a5445adb4cd168e0b459037e8405cf5d5b256edb82

Scenarijus nr. 20

- Inicijavimo transakcijos maiša: 0x1d38e63fcd66243dffb643606f16e34c1b9c527da147ab5d296499b8adc3f39a
- Keitimo transakcijos maiša: 0x95e2b43572146ee287095df6c5ba13a10e5d17c8571c73d3ed42c7569af3c01e
- Atakos transakcijos maiša: 0xef2d3d316c95ac5e788b8cb618d9bb5ad767c5de6b7fb884837412f08dbd1c3b

Scenarijus nr. 21

- Inicijavimo transakcijos maiša: 0xd4403d9be32dff48eb356e902e1a441e10d52473a0cf7287a8a7d75133d5a49f
- Keitimo transakcijos maiša: 0xf9b0e05a60fbec168b0db97acf4cb4cdeed8de08fa4d1a1b09137a1fbd3d77a4
- Atakos transakcijos maiša: 0x3e92ad9269b5c908279b408c05a776ed46b37636495fe339ea5b14d5b1b0e161

Scenarijus nr. 22

- Inicijavimo transakcijos maiša: 0x7b8c653c1017fb9734996d892327f1af52ba4353cca5e6f6d2ba1d57168a21ac
- Keitimo transakcijos maiša: 0x4c1cf46dc38807a291256e2573b06e8f407b75ff9e688c7f9d23a111012bb1b5
- Atakos transakcijos maiša: 0xe3ff3155802d00c2484a08ebdc5aaeea00a25c829da30f9a111d6db67515c64d

Scenarijus nr. 23

- Inicijavimo transakcijos maiša: 0xe11ebd4dd93b8b5495090d9f8406583ac0f205d393c06ab4fd1f50384e9f656f
- Keitimo transakcijos maiša: 0x2f9bc7b828dd07b8a2d554f6878b64deafe0e773b1424e32ccc7b0da78ed539c
- Atakos transakcijos maiša: 0xac290a0c9376ad83486fdd813d2c151c45d87b20ae77ca0683faf141e9fd7479

Scenarijus nr. 24

- Inicijavimo transakcijos maiša: 0x9b7578e929fd900cbce548f0900426c1abfb134e14f762c99d4ec4f0c4641d5d
- Keitimo transakcijos maiša: 0x881087ca1b5385127129d6f37d3e86a119d5d78d6045bd83a755e4dd7dc65bee
- Atakos transakcijos maiša: 0x4612878842497c26ca27b1198b8135e581faa20fee39420f41a50041236c8185

Scenarijus nr. 25

- Inicijavimo transakcijos maiša: 0xf305994b4af393fb3427318a4bb94c7d8bfd6503f01ac9d9a2a4be7987b06059
- Keitimo transakcijos maiša: 0x4e7e56eacaddc8456685c3ad49864718ad5ae21b6d909cb96abc456c89c36087
- Atakos transakcijos maiša: 0x731f1fac9f69d83356677846bfff56dad97092d5ef16018df172cf8b018e98ee
- Atšaukimo transakcijos maiša (sėkminga): 0xa11ea97ae75c7a9b736c920be61407d4003355b61790beec0697e9d8e8412833

Scenarijus nr. 26

- Inicijavimo transakcijos maiša: 0x9da16fbe12038365d015566b2b51db91388abf73f1b129105a56b0de5b1fa3db
- Keitimo transakcijos maiša: 0x84c656bc7465dfe1718ed9f6c235f4e757e635f3b61e03fe0a7905217f2b0355
- Atakos transakcijos maiša: 0x18c36b23071388947ded75bab3b3a832b20e17def3a22f3d6ce2221c0775f372

Scenarijus nr. 27

- Inicijavimo transakcijos maiša: 0xe1084bc632409a2ebcc42634695fc0187ef0c8bdb93653c8fe748be747f83bef
- Keitimo transakcijos maiša: 0xde81e047b4e03990daca626b4613f11ea837e53eeb3ac63b7de74fa2800f257c
- Atakos transakcijos maiša: 0x827ffde99ea7ef2533d124b0911187a4d6a800d29eea47436e3d0c0434552679

Scenarijus nr. 28

- Inicijavimo transakcijos maiša: 0x780227d5556c2ef53403335806d8569486c32d2839123b94d6333cece5ad6c7e
- Keitimo transakcijos maiša: 0x0549caa29104d3edac28e786b0f7b9f9b681e1d48f9947f0ba0d55f372ca573a
- Atakos transakcijos maiša: 0xd4ebd9af1b88cc93d8f81cfc8e622e72ece9324cac335ac137b5fd20f4e0f689

Scenarijus nr. 29

- Inicijavimo transakcijos maiša: 0xa02b4a8a7750041ddea890147b42c2ba1ca3f3143d618799ae92e2c88eed1295
- Keitimo transakcijos maiša: 0xcfd913a2e00b0547388a3094165a9a077c4d8874ee4a349d120cc519a930e0b1
- Atakos transakcijos maiša: 0x588ef01b7fb840367a3c2532d492f28aaabef5dc47efa0d385b8e06372385d91

Scenarijus nr. 30

- Inicijavimo transakcijos maiša: 0x3e7b6773533634629f7213f0d84fb5b39b7d91790e80a095f9f86baf82fee57e
- Keitimo transakcijos maiša: 0xec0c179f65275d8d6f2d6859b37068e08daaa60e3d593d903f856a8606b7191a
- Atakos transakcijos maiša: 0xaa780078d403068685c6d01754aa73d0a06776a7e97f026f03980d1666acc5d9
- Atšaukimo transakcijos maiša (sėkminga): 0x5e9eab14fd6937039047ff4f9de58b34e5260ee8f939ac9f43fde7bc9b2b6255

IŠANKSTINĖS SANDORIŲ SIMULIACIJOS ATAKŲ NUSTATYMO METODO TESTAVIMO TRANSAKCIJOS

Pateikiamos testavimo metu įvykusių transakcijų maišos ir, jei vyko atšaukimo transakcija, sėkmės rezultatas. Nepateikta atšaukimo transakcijos maiša scenarijuose indikuoja, jog ataka nebuvo nustatyta.

Scenarijus nr. 1

- Inicijavimo transakcijos maiša: 0x5022dbd0c62d5f739fd8b152fc1eb77cfb0d082a588e9db7a04f2b879f4a7c62
- Keitimo transakcijos maiša: 0xc69c3ee77182bc167388c1a4c6e804d057448349e8cf384d9430ce80a95933f0
- Atakos transakcijos maiša: 0x893ce7fc28852eecf03f16ab680550b3bbcfc102474a60f4122fd11c2dd9259c

Scenarijus nr. 2

- Inicijavimo transakcijos maiša: 0x3b19499a8d504d1703134a852278c5f3bc926624ae61fb2ab2727af7fcf7d18a
- Keitimo transakcijos maiša: 0x41530da155c6f19b5ff4803dc0dd79c475b0e8217dd6955d45a4d57a6d883429
- Atakos transakcijos maiša: 0xcc7eb4ef41ff51b29ba13268ce62fa353eddf740b163b57f9003239c201e85c4
- Atšaukimo transakcijos maiša (sėkminga): 0x1d56148df9f8167ab6f7bb7062ad8601f9cf97690891b960c70bfbe95cf17a26

Scenarijus nr. 3

- Inicijavimo transakcijos maiša: 0x3bd3b6906de524ab9bd5ad23047355554a956694d785165f60d30d7cc9413da1
- Keitimo transakcijos maiša: 0x420b86ff99154fb8933ba41857a735a2724a8cc179f22f5b243252c05bf5ea09
- Atakos transakcijos maiša: 0x4b99f285604a9e48bc773d402bcacab59e08ac578135f80e3c0d649447e0b3f8

Scenarijus nr. 4

- Inicijavimo transakcijos maiša:
  0x2b63e6162352a94f3bbd8e4304c2cbae0808cee5a4509ac25cd4056216f9acc5
- Keitimo transakcijos maiša:
  0x91be47e0ad70c530cf4785e84e7e9cdff7a213ca151b978df55fc658ef65d58a
- Atakos transakcijos maiša:
  0x97c8620700ffa13889a3754983c4594abffdb77a4b084f7ee86c57c6dc9f4acb
- Atšaukimo transakcijos maiša (nepavykusi):
  0x8547be583feda90d788c93cbfd01f5390caa5cbbb8233abefe30db6b32549685

Scenarijus nr. 5

- Inicijavimo transakcijos maiša:
  0x91be47e0ad70c530cf4785e84e7e9cdff7a213ca151b978df55fc658ef65d58a
- Keitimo transakcijos maiša:
  0x4fde31d60ea2d50104c6162271e186e056dd64ce9b1395b901b3c1b526dc0e4d
- Atakos transakcijos maiša:
  0x477d792459e8e2003e375f146cf3a3391fa4b52d50ca53452e49c6dc75fb22c0

Scenarijus nr. 6

- Inicijavimo transakcijos maiša:
  0xc22ed9eb4810b380132b9125193729b989345556ae89e6182c4fb3b687e84e9f
- Keitimo transakcijos maiša:
  0x9a80608ec5204affdf602e8758afc02debe0d8ee82b3bf7bdfbbb2a70d6f6a67
- Atakos transakcijos maiša:
  0xd23f999108571ee0af4f15ce8c2d3faa772f87bf0a404220116f645e66855021
- Atšaukimo transakcijos maiša (sėkminga):
  0x0e83f0ba1fd44900bd6425ebfe6326f888d1b389fadd6ec05860765a22a1d177

Scenarijus nr. 7

- Inicijavimo transakcijos maiša:
  0x6442a08999096443bfb51e55c6f028ed6841f80ac028ef73dc4944a4329bc052
- Keitimo transakcijos maiša:
  0xea9c71f9ff2a9740ac8a12807bace0110721a4a188429129a857a29d5728a8f7
- Atakos transakcijos maiša:
  0x0941e996c50c0c1831579ec46c3e3c0f6ee106779dfc1d086a28d969c936ae82

Scenarijus nr. 8

- Inicijavimo transakcijos maiša:
  0x6452262eb8d85a75dfa37d02101fd946a3554bc15abae903d628fcaa572d56f0
- Keitimo transakcijos maiša:
  0xd71d2291a16edbe36790dfc02af482d74c6f59148e40f6d128c930ad52940230
- Atakos transakcijos maiša:
  0x3e34b64f5d3c16e4c1a97ca59036e35db3c5116eb204e96321b99fce0a7117e9
- Atšaukimo transakcijos maiša (sėkminga):
  0xd60901f2107e005f47cf72cbb068f0a24c059917bd59be2aa2bb9bc01bdee280

Scenarijus nr. 9

- Inicijavimo transakcijos maiša:
  0xd71d2291a16edbe36790dfc02af482d74c6f59148e40f6d128c930ad52940230
- Keitimo transakcijos maiša:
  0x12100293b224be7c9c10da4152fb19576688bae0518e995fadfab884289bb1e3
- Atakos transakcijos maiša:
  0x8780528dde26f48b9ce23965f6307ecc88d79bffbac3b2f62ab019259717287e

Scenarijus nr. 10

- Inicijavimo transakcijos maiša:
  0x5ec381486d645147dadaf11aedac4b62dc522bd7c8dbe1458e98a035f1c2d2a8
- Keitimo transakcijos maiša:
  0x05e1ebb333174490dfccb0f2ce6d5cbaf2759590c7af3a2d21b0d01c09d827e0
- Atakos transakcijos maiša:
  0x5b82fc77b8d0769bf2ad620e40ac82974835959e82c4ddcfbc6c0aae7f38cc96
- Atšaukimo transakcijos maiša (sėkminga):
  0x2a1a22746a259f6d424c787fb7cbecb6276034868564c5a3b61321f8a4171606

Scenarijus nr. 11

- Inicijavimo transakcijos maiša:
  0x8da4d43fff52e1bc57206a1f3b6197229731805732f9a756665b22001c24b092
- Keitimo transakcijos maiša:
  0x1d35eed85a7c4873bd6c5ef6f70fcae41848b33a7a45e04e548032763c1e034a
- Atakos transakcijos maiša:
  0x0533a63990cba9da437e88197958fec3858f0187b41705b5470e833a11cdabd4
- Atšaukimo transakcijos maiša (sėkminga):
  0x8c3406b6614f4b211e292c01725d9a382176ccd4890f3fbbb6ed911b60cb77ad

Scenarijus nr. 12

- Inicijavimo transakcijos maiša:
  0x4e8a4290c2790cc9ae85a654ad0863478917c89719db579639833dc48565d134
- Keitimo transakcijos maiša:
  0x7a60e535ba24ef71e18b0bdc2f244b0f627eece9b803bfdaa7d54e00fbdd08b9
- Atakos transakcijos maiša:
  0xbf02e8e565ba3eacfd8777012673d63df74d9ba657b6a41e7360a2251b2369ed

Scenarijus nr. 13

- Inicijavimo transakcijos maiša:
  0x1d311de59d33b7159b97edfd085e7f838820dd4d6e02d230001c34110de524f8
- Keitimo transakcijos maiša:
  0xe987123bd88b658e4cc73a898080b3fa18305f122ad9835fd35a4a2df8ce23f7
- Atakos transakcijos maiša:
  0xc5ed7fe6ad845ee49091f2c5b123dae73bec3297a43881b6d63c8ee9da90d61d

Scenarijus nr. 14

- Inicijavimo transakcijos maiša:
  0xf3b890e81cd04a12ef4831431f38873b42a5f91cdfc7ce0414e646b4ec75face
- Keitimo transakcijos maiša:
  0xd22ce0fe5508c1e9b5d2d7dd9ce017a95d6e7065f3292770956ebd2de9bb4bb7
- Atakos transakcijos maiša:
  0xab9b9891e54483b986b43cd6b362afddb5e55296d0a57003254c273633b0101e

Scenarijus nr. 15

- Inicijavimo transakcijos maiša:
  0x17c92d56b8f51407d35ada2f7e0341ddb9ca7cca7f3a5a532859cb6ed81f3e5f
- Keitimo transakcijos maiša:
  0x3e049ab99179070d4afb28e2a8f4a24722ef6e9f8d0e541cc219bd08141cec95
- Atakos transakcijos maiša:
  0xc7e2dab6e680543f12d0721d37d9fd3e37c8736e61c69f4177a72684620f3d72

Scenarijus nr. 16

- Inicijavimo transakcijos maiša:
  0x96ae683865f2992dd4a8eeb6bd24f00a3d67dbc0641045faa3dd29d9717ed0ca
- Keitimo transakcijos maiša:
  0xa89a2760362c3580332e4c3c6ff3dc1dedc0a82123d7740ebedb66fa7c05528c
- Atakos transakcijos maiša:
  0xb88acaf218a6779ab15acc97252378cf5d51687b003ea7bbf44e26640ac1d24d

Scenarijus nr. 17

- Inicijavimo transakcijos maiša: 0xf4a6fe9a94ca855ad95ec1c6bb88b040721b27eec4dade6467ac558e12fb47b6
- Keitimo transakcijos maiša: 0x8592e1170f512fa33f080422a1119c18df92f656d34e6cad35d3cda0b2bba705
- Atakos transakcijos maiša: 0xef86addb5eda0ea9ea7e3944fbb7cb3666c48bd464416817d7e605d92d6bde32

Scenarijus nr. 18

- Inicijavimo transakcijos maiša: 0xc2e76b9fe8750f5b76a733f2fa174630c2fa5be777f72e972ac4d3f67fd49460
- Keitimo transakcijos maiša: 0x791bee5afd9c564fc5389329feffcb582d0a8d4d223e1e1239358a215ad746b1
- Atakos transakcijos maiša: 0xe35ffb0157a9682b2d85696673b180d396a4cdf9fe44531e1590b619516bce15

Scenarijus nr. 19

- Inicijavimo transakcijos maiša: 0xb72f22c4398897e691f4fbe96a9de73afa19827baf69bfc5c62f1c4606405a0b
- Keitimo transakcijos maiša: 0x5fbe508c46dfb6f6f54310c0c5e454cb87f480a1e245d8c939067c5edd5dd110
- Atakos transakcijos maiša: 0xe8f1fefb618b21e17612ff154af5325878037707433a1ef4d7205f9ca528ed78

Scenarijus nr. 20

- Inicijavimo transakcijos maiša: 0x525704476acc9f355117bc447b980bc65d49c4391b889bb69b06c13f61e2a63d
- Keitimo transakcijos maiša: 0x9eb2d9becf10286107523ceb6559928aac9d60e90dfd3a8121bc5d6c5ac4ba1d
- Atakos transakcijos maiša: 0x357f5ebe72f102c2dd34cc8b4b0db26479121534a27ed4f8e90cd7b3f53fb400

Scenarijus nr. 21

- Inicijavimo transakcijos maiša:
  0xd8e86411ab00b23886cd3835233bee66164321e8461612ec32035eed1bd8253e
- Keitimo transakcijos maiša:
  0x384e78b75a997492e1355e9c0a5209a324dfef6626828b89fbfb2ee184dbc137
- Atakos transakcijos maiša:
  0x218c2f9d6252b42ef5d70066857510f02b9fd61c0724d4b2b9d0808f23166e2b
- Atšaukimo transakcijos maiša (nepavykusi):
  0xd130da005eb3a10bc823cc1166400218215be47ec616fb1594712a1bb0e72c9c

Scenarijus nr. 22

- Inicijavimo transakcijos maiša:
  0x85a3a614ad3ac643509ec075f5d1a79452f66cb0326bdbfab4106c008c7c7982
- Keitimo transakcijos maiša:
  0xc4e18d71a34495068ecb15ccd20e850ef26e915bc53a8bbdf5f67de78fb531c9
- Atakos transakcijos maiša:
  0x52e6c9a70c7538c74439b6520979f6421b77aa4ae93244204c34fe1d07e9feae
- Atšaukimo transakcijos maiša (nepavykusi):
  0x1898c717e85e62274bbebcd950fcacb4c0383fe54853b695239fdb51247ca908

Scenarijus nr. 23

- Inicijavimo transakcijos maiša:
  0x14d3fbdb9feb579a77886466f478427a80e6cfe7dee3ec4201d7ed493bfcdbc8
- Keitimo transakcijos maiša:
  0x10afbc413a643cb36ca6e4a4a45dd34c721a2a2317035f497d5bcf9694444598
- Atakos transakcijos maiša:
  0xd0a3ee3f2bdc0269ff7db3a4ac899bebdd4f2f881517148c93fd6a9d83cba5ff

Scenarijus nr. 24

- Inicijavimo transakcijos maiša:
  0xfbac037118dd287cdf5be70861a6e50a515139412a978a1b2524536fe4e646d4
- Keitimo transakcijos maiša:
  0xbbc29cc091ade5d906a47ad8dd38da9057b4f8f1b25379caebf3c25017218f6e
- Atakos transakcijos maiša:
  0xb82bde7b99180f45fbe346f5d8bcf9cba013d2bdfc733b2f162fda48d96af584

Scenarijus nr. 25

- Inicijavimo transakcijos maiša:
  0x3ab782c12381dfbfae9f5f8271c57400c59181541e91ca5c58a6d82cf6976559
- Keitimo transakcijos maiša:
  0x155c1688050382a6827a017a8c747ae8bcd47d8785571b0bc5cb8c824f15e543
- Atakos transakcijos maiša:
  0xe1234b0c4340740cb924ed3a8f8b44d381395fc835b56d0dd203afea881b2ee0

Scenarijus nr. 26

- Inicijavimo transakcijos maiša:
  0x961e409647da0e3389f4f4085a8e89a918245fd95a8161b0866f36b63bb719b1
- Keitimo transakcijos maiša:
  0xce210cf74762914fcc43a285e9f1e8e8ffab61190b425750008f5e658423a507
- Atakos transakcijos maiša:
  0x34fc2d0353510ac9c7307b2245ef78abff2e5e33f0ddcf218d26e37198186ee8
- Atšaukimo transakcijos maiša (sėkminga):
  0xe6c62e24b98841b6f382268b63d1f7671fbf0b74ac0aba84839dfe3862d3b3f1

Scenarijus nr. 27

- Inicijavimo transakcijos maiša: 0xd3570a506927ba890ba4f166107f95a2088666b7c0041c4f6568a111272228db
- Keitimo transakcijos maiša: 0x3df20420c02b24d9bdb37aff24fdd3523001dce6df125895821e46ebd4620af2
- Atakos transakcijos maiša: 0xb8157142d6e6a71a43d7732389477b0239219d28427ec95a1f27a707619b9288
- Atšaukimo transakcijos maiša (sėkminga): 0xbf7bd39470d545b12e6d9710d1898314f36d394aebb378304df3f083abdaac38

Scenarijus nr. 28

- Inicijavimo transakcijos maiša: 0xbd0401260ad125f2865dd821ee9e8c29e4bed32b87497690f62d73b9f8f22447
- Keitimo transakcijos maiša: 0x95dfd72d52b3c8ed4f55f1d5a1d34b37a2433144e25c8e3bfafe918a7b27fca3
- Atakos transakcijos maiša: 0xdf7aa7d6137b129d71b96805af41d90ea71eaab01c59d85238dd9462f8ba618f

Scenarijus nr. 29

- Inicijavimo transakcijos maiša: 0xe0f9073a7d5ab653b55157d8015df70eb613fe55da00fd05c046d188603662b6
- Keitimo transakcijos maiša: 0x8fd2fdaca60d611fd38f5edc792b9af44dcf83ea7b3c0e982e86ac6fd0cb99e5
- Atakos transakcijos maiša: 0xf2c31b887418311bce81996eb2724d5a320cfc1da2bbac4f58231113a25fd586

Scenarijus nr. 30

- Inicijavimo transakcijos maiša: 0x5207546752fe92b24181a83a83c37db19732112130fd082a404e3f977596bd1f
- Keitimo transakcijos maiša: 0x8933d0aba81bca0de9c188bcff03d6c89770cd97efd0bd0e955e145525401c5a
- Atakos transakcijos maiša: 0x985777e4255f5cec9567704db09c2a1d73afd51151ab023803db493dd361e191