KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

Tautvydas Traška

Saugus informacijos saugojimas naudojant blokų grandinių technologijas

Baigiamasis magistro darbas

Vadovas
prof. Algimantas Venčkauskas

KAUNAS, 2020

KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

KOMPIUTERIŲ KATEDRA

Saugus informacijos saugojimas naudojant blokų grandinių technologijas

Baigiamasis magistro darbas

Informacijos ir informacinių technologijų sauga (kodas 6211BX008)

Vadovas
(parašas) prof. Algimantas Venčkauskas
(data)

Recenzentas
(parašas) doc. Nerijus Morkevičius
(data)

Projektą atliko
(parašas) Tautvydas Traška
(data)

KAUNAS, 2020

KAUNO TECHNOLOGIJOS UNIVERSITETAS

Informatikos fakultetas

(Fakultetas)

Tautvydas Traška

(Studento vardas, pavardė)

Informacijos ir informacinių technologijų sauga, 6211BX008

(Studijų programos pavadinimas, kodas)

„Saugus informacijos saugojimas naudojant blokų grandinių technologijas“

AKADEMINIO SĄŽININGUMO DEKLARACIJA

20 ___ m. _____ ____ d.

Kaunas

Patvirtinu, kad mano **Tautvydo Traškos** baigiamasis projektas tema „Saugus informacijos saugojimas naudojant blokų grandinių technologijas“ yra parašytas visiškai savarankiškai, o visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

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

(vardą ir pavardę įrašyti ranka)	*(parašas)*

Traška, T. „Saugus informacijos saugojimas naudojant blokų grandinių technologijas“. Magistro baigiamasis projektas / vadovas prof. Algimantas Venčkauskas; Kauno technologijos universitetas, informatikos fakultetas, kompiuterių katedra.

Mokslo kryptis ir sritis: fiziniai mokslai, informatikos kryptis.

Reikšminiai žodžiai: saugumas, informacija, saugojimas, Blockchain.

Kaunas, 2020. 86 p.

SANTRAUKA

Šiame darbe analizuojama blokų grandinių technologija ir jos pritaikymas saugiam informacijos saugojimui. Darbas suskirstytas į keturias dalis: analizė, projekto aprašas, prototipo realizacija ir prototipo tyrimas. Kiekvieno skyriaus pabaigoje pateikiamos išvados. Prieduose pateikiami sukurto prototipo tyrimo rezultatai.

Analizės dalyje analizuojami įvairūs informacijos saugojimo metodai, jų problemos ir charakteristikos. Toliau analizuojama blokų grandinių technologija pažvelgiant į veikimo principus ir panaudojimo atvejus. Apžvelgiami blokų grandinių tipai ir atliekamas jų palyginimas. Aprašomi blokų grandinių pranašumai saugiam informacijos saugojimui lyginant su įprastais metodais. Analizė apibendrinama išvadomis.

Antrame skyriuje aprašomas sprendimas saugiam informacijos saugojimui naudojant blokų grandinių technologiją remiantis atliktos analizės išvadomis. Pateikiama sprendimo metodo koncepcija ir aprašomas veikimo principas. Surinkti ir aprašyti panaudos atvejai, funkciniai ir nefunkciniai reikalavimai. Aprašytas blokų grandinės veikimas kuriamame sistemos modelyje. Išvardintos ir detaliai aprašytos failų sistemos operacijos. Pateikiamas statinis bei dinaminis sistemos vaizdas ir sprendimas sistemos saugumo realizacijai atlikti. Siūlomas sistemos modelis apibendrinamas išvadomis.

Trečiame skyriuje išdėstytas kuriamos sistemos prototipo realizacijos detalės. Išvardintos panaudotos darbo priemonės ir technologijos, aprašyta architektūra ir komponentai. Aprašomas duomenų modelis, saugumo realizacija ir sistemos būsenos. Prototipo realizacija apibendrinama išvadomis.

Realizuotas prototipas ištirtas kokybiškai ir kiekybiškai. Tyrimo metu išryškintos siūlomo metodo charakteristikos. Visi šie rezultatai pateikti ketvirtame skyriuje. Kiekybinio tyrimo rezultatai pridėti dokumento priedų dalyje.

Darbas apibendrinamas galutinėmis išvadomis dokumento pabaigoje.

Traška, Tautvydas. *Secure Information Storage Using Blockchain Technology*: *Master's* thesis in Department of Computer Sciences / supervisor assoc. prof. Algimantas Venčkauskas. The Faculty of Informatics, Kaunas University of Technology.

Research area and field: physical sciences, informatics.

Key words: secure, information, storage, Blockchain.

Kaunas, 2020. 86 p.

SUMMARY

The purpose of this thesis is to analyze Blockchain technology and find a solution for secure information storage using it. The work is divided into four main chapters: subject analysis, the suggested method of project implementation, realization of prototype and prototype testing. Conclusions are presented at the end of each chapter. The appendices present the results of the research of the developed prototype.

The analysis part analyzes various information storage methods, their problems, and characteristics. Blockchain technology is further analyzed in terms of operating principles and applications. The types of Blockchain are reviewed and compared. The advantages of Blockchain for secure information storage compared to conventional methods are described. The analysis is summarized in conclusions that lead to the development of further chapters.

The second chapter describes a solution for secure information storage using Blockchain technology based on the findings of the analysis. The concept of the solution method is presented, and the principle of operation is described. Use cases, functional and non-functional requirements are collected and described here. The operation of the Blockchain in the system model is described as well. File system operations are listed and described in detail. A static and dynamic image of the system and a solution for the implementation of system security are presented. The proposed system model is summarized in conclusions.

The third chapter sets out the details of the prototype implementation of the developed system. The tools and technologies used are listed, the architecture and components are described here. Later in this chapter the data model, security implementation, and system states are presented. The realization of the prototype is summarized in conclusions.

The realized prototype was analyzed, and its performance was tested. The testing highlighted the characteristics of the proposed method. All the results are presented in Chapter Four. The testing results are attached in the annexes of the document as well.

The work is summarized at the end of the document and final conclusions are drawn.

TURINYS

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

TERMINŲ IR SANTRUMPŲ ŽODYNAS

ACL (angl. Access Control List) – prieigos kontrolės sąrašas, susijęs su kompiuterine failų sistema, kai prie objekto pridedamas teisių sąrašas.

AES (angl. Advanced Encryption Standard) – simetrinio šifravimo algoritmas.

AIFF (angl. Audio Interchange File Format) – garso failo formato standartas, naudojamas asmeninių kompiuterių ir kitų elektroninių garso įrenginių garso duomenims saugoti.

APFS (angl. Apple File System) – patentuota failų sistema „MacOS High Sierra“ (10.13) ir naujesnėms macOS versijoms.

BDAR – bendrasis duomenų apsaugos reglamentas.

Blockchain (liet. Blokų grandinė) – didėjantis įrašų, vadinamų blokais, sąrašas, susietų naudojant kriptografiją.

Btrfs – failų sistema, pagrįsta copy-on-write (COW) principu sukurta naudoti „Linux“ operacinėje sistemoje.

Cloudian – JAV duomenų saugojimo įmonė.

DDoS (angl. distributed denial-of-service attack) – kibernetinė ataka, kai auką užplūsiantis srautas gaunamas iš daugelio skirtingų šaltinių.

Dell EMC – amerikiečių daugianacionalinė korporacija, kurios būstinė yra Hopkintone.

DOS (angl. Disk Operating System) – nuo platformos nepriklausomas disko operacinės sistemos akronimas, kuris tapo įprastu disko pagrindu veikiančių operacinių sistemų, suderinamų su IBM PC, santrumpomis.

ECDSA (angl. Edwards-curve Digital Signature Algorithm) – elipsinės kreivės skaitmeninio parašo algoritmas.

eSATA (angl. external Serial AT Attachment) – SATA variantas, skirtą išoriniam sujungimui.

EXT2 (angl. second extended file system) – išplėstinė failų sistema „Linux“ branduoliui (antra versija).

exFAT (angl. Extensible File Allocation Table) – failų sistema, optimizuota „flash“ atminčiai, pavyzdžiui, USB atminčiai ir SD kortelėms.

EXT (angl. extended file system) – išplėstinė failų sistema „Linux“ branduoliui (pirmoji versija).

EXT3 (angl. extended file system) – išplėstinė failų sistema „Linux“ branduoliui (trečioji versija).

EXT4 (angl. extended file system) – išplėstinė failų sistema „Linux“ branduoliui (ketvirtoji versija).

FAT (angl. File Allocation Table) – failų sistemos architektūra ir ją naudojančių pramonės standartinių failų sistemų šeima.

FAT-12 (angl. File Allocation Table) – failų sistemos architektūra, naudojanti 12 bitų ilgio adresus.

FAT-16 (angl. File Allocation Table) – failų sistemos architektūra, naudojanti 16 bitų ilgio adresus.

FAT-32 (angl. File Allocation Table) – failų sistemos architektūra, naudojanti 32 bitų ilgio adresus.

GDPR (angl. General Data Protection Regulation) – Bendrasis duomenų apsaugos reglamentas.

IBM (angl. International Business Machines Corporation) – amerikiečių daugianacionalinė technologijų įmonė, kurios būstinė yra Armonke, Niujorke.

IDE (angl. Integrated Drive Electronics) – Pirmoji ATA/ATAPI sąsajos versija.

Java – objektinio programavimo kalba.

JFS (angl. journaling file system) – žurnalinė failų sistema.

JPEG – dažniausiai naudojamas skaitmeninių vaizdų, ypač skaitmeninės fotografijos, glaudinimo būdas.

JRE (angl. Java Runtime Environment) – Java programų vykdymo aplinka.

Kerberos – kompiuterinio tinklo autentifikavimo protokolas.

Linux – atvirojo kodo „Unix“ tipo operacinių sistemų šeima, pagrįsta „Linux“ branduoliu.

macOS – patentuotų grafinių operacinių sistemų, sukurtų ir parduotų „Apple Inc.“, serija.

MD5 (angl. message-digest algorithm) – pranešimų santraukos algoritmas sukuriantis 128 bitų maišos vertę.

MinIO – debesų saugyklos serveris, suderinamas su „Amazon S3“.

NoSQL – duomenų bazė, sumodeliuota kitomis priemonėmis nei lentelių struktūra, naudojama reliacinėse duomenų bazėse.

NTFS (angl. NT File System) – patentuota žurnalinė failų sistema, kurią sukūrė „Microsoft“.

PNG (angl. Portable Network Graphics) – grafikos failo formatas, palaikantis duomenų glaudinimą be nuostolių.

RAID (angl. Redundant Array of Independent Discs) – duomenų saugojimo virtualizacijos technologija, sujungianti kelis fizinio disko įrenginio komponentus į vieną ar kelis loginius vienetus duomenų dubliavimo ir veiklos tobulinimo tikslais.

RBAC (angl. role-based access control) – rolėmis grįstas apsaugos būdas apriboti sistemos prieigą įgaliotiems vartotojams.

Red Hat – amerikiečių daugianacionalinė programinės įrangos įmonė.

RSA (angl. Rivest–Shamir–Adleman) – viena iš pirmųjų viešojo rakto kriptosistemų plačiai naudojama saugiam duomenų perdavimui.

SATA (angl. Serial AT Attachment) – kompiuterinės magistralės sąsaja, jungianti pagrindinės magistralės adapterius su atminties įrenginiais, tokiais kaip kietasis diskas.

SCSI (angl. Small Computer System Interface) – standartų rinkinys, skirtas fiziškai sujungti ir perduoti duomenis iš kompiuterių į periferinius įrenginius.

SHA1 (angl. Secure Hash Algorithm 1) – kriptografinė maišos funkcija, kuri priima įvestį ir sukuria 160 bitų (20 baitų) maišos vertę.

SNIA (angl. Storage Networking Industry Association) – pelno nesiekianti prekybos asociacija, kurios misija – vadovauti saugyklų pramonei visame pasaulyje.

Solaris – patentuota „Unix“ šeimos operacinė sistema, kurią sukūrė „Sun Microsystems“.

SQL (angl. Structured Query Language) – struktūrizuota užklausos kalba, naudojama programuojant ir skirta valdyti duomenis, laikomus reliacinių duomenų bazių valdymo sistemose.

SQL injekcija – ataka, kai yra įvedamos kenksmingos SQL komandos į neapsaugotus duomenų įvedimo laukus.

SSD (angl. solid-state drive) – atminties įrenginys, naudojamas nuolatiniams duomenims saugoti, paprastai naudodamas „flash“ atmintį.

SUSE – vokiečių kilmės daugianacionalinė atvirojo kodo programinės įrangos įmonė, kurianti ir parduodanti „Linux“ produktus verslo klientams.

TCP (angl. Transmission Control Protocol) – vienas iš pagrindinių interneto protokolų.

TIFF (angl. Tagged Image File Format) – kompiuterio failo formatas, skirtas saugoti grafikos vaizdus.

TLS (angl. Transport Layer Security) – protokolas, skirtas komunikacijos saugai užtikrinti tinkle.

UFS (angl. Unix file system) – failų sistema, palaikoma daugelio „Unix“ šeimos operacinių sistemų.

USB (angl. Universal Serial Bus) – pramonės standartas, nustatantis kabelių ir jungčių bei protokolų specifikaciją, skirtą ryšiui ir energijos tiekimui tarp kompiuterių ir periferinių įrenginių.

WAV (angl. Waveform Audio File Format) – garso failo formato standartas.

Windows – patentuotų grafinių operacinių sistemų šeima, kurias sukūrė ir parduoda kompanija „Microsoft“.

XFS – didelio efektyvumo 64 bitų žurnalinė failų sistema.

XML (angl. Extensible Markup Language) – žymėjimo kalba, apibrėžianti dokumentų kodavimo taisyklių rinkinį, suprantamą žmonėms ir kompiuterių sistemoms.

ĮVADAS

Magistro baigiamasis darbas tema „Saugus informacijos saugojimas naudojant blokų grandinių technologijas“ priklauso „Informacijos ir informacinių technologijų sauga“ studijų programai. Darbe analizuojami saugaus informacijos saugojimo metodai, apžvelgiamos jų savybės ir problemos. Analizuojama blokų grandinių technologija ir jos galimybė pritaikymui kuriant saugų informacijos saugojimo metodą.

Sparčiai besivystančiame pasaulyje duomenų apsauga tampa vis svarbesne problema, todėl reikia nuolat ieškoti geresnių sprendimų informacijai saugoti. Projektuojant informacijos saugojimo metodą, didžiausias dėmesys skiriamas trims pagrindiniams aspektams:

- konfidencialumas;
- vientisumas;
- prieinamumas.

Darbe išryškinama kiekvienos dalies svarba. Pagal apibrėžtas problemas ir uždavinius projektuojamas informacijos saugojimo metodas naudojant blokų grandinių technologiją. Kuriamas sprendimas skirtas bet kokio tipo naudotojams, kurie yra suinteresuoti saugiu informacijos saugojimu tinkle.

Darbo problematika ir aktualumas

Informacijos saugumas yra nepaprastai svarbus tiek fiziniams asmenimis, tiek verslui. Duomenys apie finansus, verslą, produktus ir bet kokia kita informacija turi būti patikimai apsaugoti. Žmonių skaičius pasaulyje auga ir smarkiai plečiasi technologijų naudojimas įvairiose srityse. Kartu su greitu pasaulio tempu juda duomenų saugojimo ir apsaugos technologijos. Esamiems saugojimo sprendimams nuolat atrandamos saugumo spragos. Todėl šio darbo metu siekiama panaudoti sparčiai populiarėjančią blokų grandinių technologiją ir pritaikyti ją saugiam informacijos saugojimo metodui kurti.

Darbo tikslas ir uždaviniai

Darbo tikslas: išanalizuoti informacijos saugojimo metodus ir pasiūlyti naują duomenų saugojimo ir apsaugos metodą naudojant blokų grandinių technologiją.

Iškelti uždaviniai:

1. išanalizuoti esamus informacijos saugojimo metodus;
2. išanalizuoti blokų grandinių technologiją;
3. pritaikyti blokų grandinių technologiją kuriant naują informacijos saugojimo metodą;
4. realizuoti veikiantį prototipą pagal pasiūlytą sprendimo metodą;
5. atlikti realizuoto prototipo kokybinį ir kiekybinį tyrimus;
6. pagal gautus rezultatus apibendrinti darbą ir pateikti išvadas.

Darbo rezultatai ir jų svarba

Darbo metu pasiūlytas saugaus informacijos saugojimo metodas naudojant blokų grandinių technologiją. Tai sparčiai populiarėjanti ir besiplečianti technologija, kurią galima pritaikyti įvairiose srityse, taip pat ir saugiam informacijos saugojimui, ir tai įrodoma šio darbo metu. Pagal pasiūlytą metodą realizuotam prototipui atliktas tyrimas, išskirtos metodo charakteristikos teikiančios naudą ir patobulinimą saugiam informacijos saugojimui.

Darbo struktūra

Darbas padalintas į keturias pagrindines dalis: analizė, projektas (metodo siūlymas), prototipo realizacija, sukurto prototipo tyrimas. Analizės dalyje analizuojami informacijos saugojimo metodai ir jų charakteristikos. Taip pat analizuojama blokų grandinės technologija, palyginami grandinių tipai, išskiriamos savybės, kurios bus naudingos kuriant informacijos saugojimo metodą naudojant šią technologiją. Antrojoje dalyje aprašomas siūlomas sprendimo metodas saugiam informacijos saugojimui. Trečiame skyriuje išdėstytas kuriamos sistemos prototipo realizacijos detalės. Išvardintos panaudotos darbo priemonės ir technologijos, aprašyta architektūra ir komponentai. Ketvirtame skyriuje aprašomas realizuoto prototipo tyrimas. Tyrimo metu išryškintos siūlomo metodo charakteristikos. Darbas apibendrinamas išvadomis.

1. INFORMACIJOS SAUGOJIMO METODŲ ANALIZĖ

Duomenų apsauga yra dalis informacinių technologijų saugumo srities, kuri nagrinėja būdus ir metodus, skirtus duomenų apsaugai įvairiose sistemose ir įrenginiuose. „Storage Networking Industry“ asociacija (SNIA) siūlo tokį sąvokos apibūdinimą: tai fizinių, techninių ir administracinių priemonių taikymas, siekiant apsaugoti informacijos saugyklas ir infrastruktūrą kartu su informacija saugoma, tokiose sistemose [1]. Duomenų apsaugos tikslas – saugoti duomenis ir saugyklos infrastruktūrą prieš neautorizuotus nuskaitymus, modifikavimus ir sunaikinimus, tuo pačiu metu išlaikant sklandžią prieigą autorizuotiems naudotojams. Tikslui pasiekti gali būti taikomos įvairios priemonės, kurios pagal savo pobūdį gali būti skirstomos į prevencines, aptinkančiąsias, korekcines, atstatančiąsias ir kitas.

Siekiant užtikrinti informacijos saugumą, reikia įvertinti tris pagrindinius aspektus, susijusius su duomenų apsauga: konfidencialumą, vientisumą ir prieinamumą. Reikia užtikrinti, kad jautri informacija nebus pasiekiama neautorizuotiems naudotojams, kad esama informacija yra patikima ir kad kiekvienu laiko momentu, duomenys būtų pasiekiami autorizuotiems naudotojams. Taip pat reikia išlaikyti finansinių kaštų balansą, nes pati apsauga neturėtų būti brangesnė už saugomus duomenis ir saugyklos infrastruktūrą.

Išskiriama keletas aspektų, dėl kurių yra nuolat stiprinama duomenų apsauga ir ieškoma naujų, efektyvesnių būdų kuriant saugumo priemones:

- Duomenų kiekio augimas. Kiekvienais metais saugomos informacijos kiekis smarkiai didėja. Saugojimui reikalingi papildomi resursai, vis didesniu informacijos kiekius sudaro svarbi informacija. Nuo kiekio priklauso ir apsaugos sudėtingumas;
- Kibernetinių atakų padažnėjimas. „Verizon“ duomenų pažeidimo tyrimų ataskaita rodo, kad 2018 m. buvo aptikta 53 000 saugumo incidentų, įskaitant ir 2 216 duomenų nutekinimo atvejų;
- Duomenų pažeidimo žalos kaina. Atstatyti žalą po duomenų nutekėjimo ar pažeidimo – nepaprastai brangu. Yra paskaičiuota, kad kompanijos vidutiniškai išleidžia apie 3,5 milijono JAV dolerių (apie 140 JAV dolerių vienam sugadintam įrašui), siekiant atstatyti patirtą žalą. Tokie skaičiai skatina daugiau dėmesio skirti informacijos saugumui;
- Didėjanti duomenų vertė. Didelių duomenų analitikų dėka, organizacijos gali greičiau įvertinti savo duomenų vertę. Atlikus analizę, įvertinami įvairūs niuansai ir pasveriamas reikiamas resursų kiekis apsaugai įdiegti;
- Reguliacijos ir nuostatai. Vyriausybės imasi priemonių ir griežtina asmenų duomenų apsaugos reguliavimą. 2018 m. paskelbtas bendrasis duomenų apsaugos reglamentas (BDAR) (angl. General Data Protection Regulation (GDPR)) verčia įmones imtis griežtesnių priemonių, siekiant apsaugoti vartotojų privatumą, o tai tiesiogiai turi įtakos informacijos saugojimo saugumo griežtinimui [1].

Kalbant apie duomenis, jų saugojimo sistemas ir pačią duomenų apsaugą visuomet galvoje turima tris pagrindines dedamąsias: konfidencialumas, vientisumas ir prieinamumas. Tai pagrindinės savybės, kurias visuomet reikia įvertinti projektuojant duomenų saugyklas. Šios savybės veikia kaip gairės įvairioms reguliavimo priemonėms. Trumpai tariant, konfidencialumas yra taisyklių rinkinys, kuris riboja prieigą prie duomenų, vientisumas – tai patvirtinimas, kad informacija patikima ir tiksli, o prieinamumas – tai garantija, kad autorizuotos šalys visuomet turės prieigą prie saugomų duomenų.

Konfidencialumas yra labai artimas privatumui. Tai reiškia, kad duomenys turi būti prieinami tik autorizuotiems asmenims. Priemonės konfidencialumui pasiekti yra kuriamos siekiant apsaugoti jautrią informaciją nuo patekimo į netinkamas rankas, tuo pačiu užtikrinant, kad informacija bus prieinama tam, kam ir priklauso.

Vientisumas reguliuoja duomenų modifikavimo procesus ir kas juos gali atlikti. Į šią savybę įeina duomenų nuoseklumo, tikslumo ir patikimumo palaikymas visu duomenų gyvavimo metu. Duomenys negali būti keičiami perdavimo metu ir turi būti imtasi visų įmanomų priemonių, kad duomenys nebūtų keičiami neautorizuotų asmenų.

Prieinamumas atsakingas už duomenų gavimą bet kuriuo reikiamu momentu. Paprastai tai pasiekiama nuolat rūpinantis aparatine įranga, užtikrinant nuolatinį veikimą ir greitus klaidos atvejų sprendimo būdus. Naudojamos įvairios priemonės tam pasiekti, pavyzdžiui, RAID (angl. Redundant Array of Independent Discs) metodika, kai į bendrą masyvą yra jungiami keli diskai, taip padidinant ne tik talpą, bet ir patikimumą bei duomenų atsarginių kopijų palaikymą. Taip pat naudojamos papildomos saugumo priemonės, tokios kaip ugniasienės (angl. firewall) ar įgaliotieji (angl. proxy) serveriai. Tai padeda apsisaugoti tiek nuo tinklo atakų, tiek nuo kenkėjiškos programinės įrangos [2].

1.1. Analizės tikslas

Analizės metu siekiama išanalizuoti blokų grandinių technologiją ir palyginti ją su tradiciniais informacijos saugojimo būdais pagal duomenų apsaugojimo aspektą.

1.2. Informacijos saugojimo metodų analizė

Informaciją galima saugiai saugoti tiek fizinėse laikmenose savo namuose, tiek debesyse (nutolusiose fizinėse talpyklose). Vienas ir kitas būdas turi savų trūkumų ir privalumų. Skiriasi patikimumas, naudojimo ir išlaikymo kaštai, apsaugos priemonės ir daugelis kitų aspektų. Aptarsime įvairius metodus ir jų siūlomas galimybes.

Optiniai diskai nuo senų laikų yra gerai žinomi kaip nešiojami, nedidelės talpos informacijos kaupikliai. Besivystant šiai technologijai, atsirado keletas skirtingų tipų diskų: CD, DVD ir Blu-ray. Įprasti CD diskai savyje gali saugoti 700 MB informacijos, DVD 4.7 GB, o Blu-ray daugiausia – 25 GB (vieno sluoksnio). Blu-ray diskai gali būti kelių sluoksnių, pavyzdžiui, dviejų sluoksnių disko talpa – 50 GB. Diske saugomos informacijos saugumas priklauso nuo to, kaip fiziškai jį laikysime. Tai kiek panašu į brangių daiktų ar pinigų laikymą seife. Jeigu norime apsaugoti diske įrašytą informaciją, privalome parūpinti saugią vietą laikymui. Taip pat diskai pasižymi savybe braižytis. Subraižytas diskas didina riziką nebenuskaityti saugomų duomenų, todėl su jais reikia elgtis itin atsargiai. Dar vienas svarbus dalykas – optinių diskų populiarumas vis labiau ir labiau krenta. Šiuolaikiniuose nešiojamuosiuose kompiuteriuose retai aptiksime optinį įrenginį diskams skaityti ar įrašyti. Asmeniniuose kompiuteriuose taip pat tokie įrenginiai dedami vis rečiau. Viskas keliama į internetą, vis daugiau žmonių turi prieigą prie tinklo, kuriame galima rasti norimą informaciją ir ją parsiųsti tiesiai į savo kompiuterį. Dėl šios priežasties diskai praranda savo populiarumą. Gali būti, kad po dešimties metų teks gerai pavargti, kol rasime įrenginį, galintį nuskaityti diske išsaugotą informaciją, todėl į juos daug investuoti nereikėtų ir vertėtų paieškoti patikimesnio ir ilgaamžiškesnio sprendimo. Nepaisant neigiamų diskų savybių, galime įvertinti tai, kad jų kaina tikrai nedidelė ir gali patenkinti tam tikrų žmonių lūkesčius. Kompanijos „Sony“ ir „Panasonic“ yra paskelbusios apie archyvams skirtus diskus, kurie gali talpinti iki 1 TB informacijos [3]. Tai tikrai įspūdingas talpos kiekis. Taip pat pranešama, kad toks diskas gali negesti iki 50-ies metų. Deja, gamyba taikoma į didesnes korporacijas ir profesionalus, o ne į eilinius vartotojus. Ir verta nepamiršti, kad bet kokį diską sugadinti lengva, todėl tai nėra pats geriausias informacijos saugojimo sprendimas. Kokybė gali priklausyti ir nuo gamintojo. Vieni žada ilgaamžį veikimą, kiti nieko nežada, o galiausiai tai – loterija. Rekomenduojama diskus laikyti vėsioje, tamsioje ir sausoje vietoje, siekiant išlaikyti jų kokybę kiek įmanoma ilgiau. Svarbu žinoti, kad į vienkartinius diskus įrašytos informacijos neištrinsime nei atsitiktinai, nei tyčia. Po įrašymo informacija negali būti užkoduota tiek apsaugos tikslais, tiek virusai negali jos keisti. Taigi, sumažėja tikimybė, kad duomenys bus pavogti. Diskus lengva klonuoti, galima turėti daug kopijų ir laikyti jas skirtingose vietose.

Kietieji diskai, kuriuos galime rasti beveik kiekviename kompiuteryje, yra naudojami ir kaip ilgalaikiai informacijos kaupikliai. Jie siūlo pakankamai gerą kainą už vieną gigabaitą ir yra gerokai greitesni už paprastus optinius diskus. Tačiau patikimumo spragos atsiranda iš jų fizinės kilmės. Tai mechaniniai įrenginiai, kuriuose veikia dideliu greičiu judančios dalys, o kartu ir sudėtinga schema su valdikliu. Kiekviena mechaninė dalis gali neatlaikyti apkrovos ir sugesti pačiu netinkamiausiu momentu. Negana to, naudojamos sąsajos, kaip ir bet kokios kitos technologijos, greitai sensta. Dabar nesunkiai galime nuskaityti eSATA ar USB palaikančius įrenginius. Tačiau problema atsiranda, jeigu

norime nuskaityti prietaisus su senomis sąsajomis, tokiomis kaip IDE ar SCSI. Lygiai toks pat likimas laukia ir šiomis dienomis naudojamų technologijų, ypač dabar, kai technologijų tobulėjimas yra įgavęs didžiulį pagreitį. Sakoma, kad kietieji diskai paprastai veikia nuo dviejų iki aštuonerių metų. Jeigu pasiseka „išspausti“ daugiau, verta nepamiršti, kad galiausiai jie visi sugenda. Praktika rodo, kad dažnesni ankstesni gedimai, nei vėlesni. Nėra jokios garantijos, todėl visuomet gera mintis atsinaujinti ir nelaikyti svarbios informacijos vienoje laikmenoje. Saugumo atžvilgiu, informacija, kuri nėra tinkamai prižiūrima, yra labiau pažeidžiama kietajame diske nei optiniame diske. Kietuosiuose diskuose informaciją keisti yra paprasta – nesunku įrašyti naują failą, pakeisti esamą ar netyčia ištrinti svarbų dokumentą. Kadangi dažniausiai kietieji diskai būna tiesiogiai prijungti prie kompiuterio, gresia virusų pavojus. Priklausomai nuo kenkėjiškos programos, mūsų informacija gali būti užkoduojama, trinama, stebima ir perduodama į išorę. Ypač gali nukentėti diskas, kuriame įdiegta kompiuterio operacinė sistema. Tinkamai neapsaugotas kompiuteris atveria kelią į duomenų vagystes ir sugadinimą. Duomenis sugadinti gali ir išorinis poveikis. Gaisro metu sudegusio disko atstatyti nepavyks. Kaip ir bet kokiam kitam duomenų saugojimo metodui, rekomenduojama turėti atsarginę kopiją (ar kelias) ir, jei įmanoma, laikyti jas saugiai ir toliau nuo pagrindinio disko.

„Flash“ atmintimi paremti kietieji diskai (angl. Solid State Drive (SSD)) yra brangesnis pasirinkimas, lyginant su tradiciniai kietaisiais diskais, tačiau tokie įrenginiai neturi judančių dalių. Tai reiškia, kad galima tikėtis ilgesnio veikimo. Technologijos nuolat kinta ir tobulėja, negalima tiksliai pasakyti kiek ilgai galima saugiai laikyti informaciją tokiame diske. Gamintojai teigia, kad jų gaminami diskai SSD diske informaciją gali išlaikyti apie dešimt metų. Tačiau atminties ląstelės SSD diskuose galiausiai praranda gyvybiškumą ir nebegali saugoti duomenų. Didelę įtaką turi įrašymų kiekis. Kuo daugiau atliekama įrašymo operacijų, tuo labiau nukenčia disko gyvavimo laikas. Žalą daro ir tai, jei paprasčiausiai disko nenaudojame. Lygiai taip pat, kaip su tradiciniais kietaisiais diskais, išlieka tikimybė, kad ateityje susidursime su jų nuskaitymo problemomis. Šiandien turime SATA sąsają, po dešimties metų viskas gali būti pasikeitę.

Nesvarbu, kokio tipo fizinę saugyklą pasirinksime, niekada negalime užtikrinti, kad duomenys bus saugūs ir juos pasiekti galėsime bet kuriuo norimu momentu. Kiekvienas fizinis įtaisas turi problemų ir rizikos išvengti negalime. Geriausias sprendimas, naudojant tokius įrenginius, nuolat daryti duomenų kopijas ir laikyti jas saugioje vietoje. Kuo labiau rūpinsimės savo duomenimis, tuo saugesni jie bus. Tvarkingas ir švarus kompiuteris taip pat gelbsti saugant informaciją, nes virusai ir kitos kenkėjiškos programos gali akimirksniu sunaikinti brangią asmeninę informaciją.

Didelę dalį problemų gali išspręsti debesija ir debesų saugyklos. Dabar gausu tiekėjų, teikiančių tokias paslaugas kaip „Google Drive“, „Dropbox“, „Microsoft OneDrive“ ar „Apple iCloud“. Tai tik vieni populiariausių rinkoje, egzistuoja daugybė kitų alternatyvių sprendimų. Šių paslaugų tiekėjai prisiima atsakomybę už duomenų saugojimą ir jų saugumą. Taip pat užtikrinamas nuolatinis pasiekiamumas, jei tik turime galimybę prisijungti prie interneto. Vienas didžiausių privalumų – mums patiems nereikia rūpintis fizinėmis laikmenomis ir jų atsarginėmis kopijomis. Nutikus nelaimei namuose, pavyzdžiui, gaisro atveju, brangi informacija išlieka saugi ir visuomet prieinama. Tai gali skambėti per gerai, kad būtų tiesa, tačiau tokios paslaugos turi savo kainą. Ilgalaikiam informacijos saugojimo sprendimui, kai kuriems žmonėms tai gali būti netinkamas sprendimas, nes tenka mokėti už paslaugos abonementą. Kuo daugiau mokame – tuo daugiau vietos gauname. Tiekėjai siūlo ir nemokamus variantus, bet to gali užtekti tik mažam informacijos kiekiui laikyti, nes, pavyzdžiui, „Google Drive“ suteikia tik 15 GB nemokamos vietos. „Apple iCloud“ siūlo automatinį duomenų surinkimą ir išsaugojimą iš turimų įrenginių. Ilgai netrunka pasinaudoti šia paslauga ir gauname pranešimą, kad nebeturime laisvos vietos. Daugumai žmonių tai neturėtų būti problema, nes 50 GB planas kainuoja vos 0,99 € per mėnesį. Taip pat yra siūlomi planai: 200 GB už 2.99 € per mėnesį ir 2 TB už 9,99 € per mėnesį. Kiekvienas gali rinktis pagal savo poreikį ir už tam tikrą kainą apsaugoti sau brangias nuotraukas bei kitą informaciją. Žinoma, tenka girdėti apie įvarius įsilaužimus ir duomenų vagystes. Taigi, kokį tiekėją rinktis ir kiek juo pasitikime, priklauso nuo mūsų pačių. Faktas tas, kad masinėse saugyklose pasirūpinta duomenų kopijomis ir egzistuoja daug mažesnė tikimybė prarasti ar sugadinti duomenis, lyginant su optiniais ar kietaisiais diskais namuose.

Saugant dokumentus ir bet kokią kitą informaciją, svarbu žinoti, kad reikia pasirinkti tinkamą formatą, kuris bus palaikomas ateityje. [3] Duomenų apsauga praranda prasmę, jeigu duomenų

negalime nuskaityti ar kitaip apdoroti. Norint kuo ilgiau išlaikyti informaciją, verta laikyti ją kuo paprastesniuose ir kuo plačiau palaikomuose formatuose. Pavyzdžiui, fotografams rekomenduojama nuotraukas saugoti originaliu neapdorotu formatu (angl. raw format) ar nesuspaustu TIFF formatu. Žinoma, tinka ir standartiniai PNG bei JPEG formatai, nes nepanašu, kad greitu metu juos kas nors galėtų pakeisti. Skaičiuoklių dokumentus (pvz.: programos „Microsoft Excel“ dokumentus) galima paruošti ilgalaikiam saugojimui, paverčiant juos į XML formato failus. Garso failus galima saugoti kuo geresnės kokybės nesuspaustu WAV arba AIFF formatu, jei tik įmanoma.

1.3. Tradicinių informacijos saugojimo metodų problemos

Įprastos duomenų bazės ir kitos saugyklos dažnai susiduria su įvairiomis problemomis ir grėsmėmis. Jas galima išskirstyti į keletą pavyzdžių:

- per didelės naudotojų privilegijos. Neatsakingai parinkus prieigos teises prie sistemos, naudotojai tiek tyčia, tiek netyčia gali pridaryti problemų modifikuodami jautrią informaciją;
- SQL injekcijos reliacinėse duomenų bazėse;
- sistemos silpnos vietos ir pažeidžiamumas;
- paskirstyta paslaugos trikdymo ataka (angl. DDoS – Distributed Denial of Service attack);
- kenkėjiškos programos;
- silpna autentifikacija;
- netinkamas atsarginių duomenų kopijų laikymas [2].

Infrastruktūros palaikymas ir priežiūra yra nemaža duomenų saugyklų problema [4]. Duomenų negalime saugoti ore, jiems reikia aparatinės įrangos ir specialistų, kurie gali ją prižiūrėti. Tai kainuoja didelius pinigus. Jeigu numatoma laikyti didelį informacijos kiekį, atitinkamai reikia paruošti ir piniginę, nes norint saugiai ir kokybiškai saugoti ir apsaugoti informaciją, teks sumokėti nemažą pinigų sumą. Saugyklų serveriams taip pat reikalinga fizinė vieta, kurioje jie bus laikomi. Kuo didesnių resursų sieksime, tuo didesnių patalpų ir pačių serverių reikės tikslui pasiekti. Tokioms problemoms spręsti verta apsvarstyti debesų saugyklas ir siūlomus jau paruoštus sprendimus kompanijų, kurių specializacija – duomenų saugojimas. Taip pat savo serverius reikia kruopščiai apsaugoti ir saugumo lygį palaikyti nuolatos atnaujinant programinę įrangą ir tinkamai ribojant prieigą prie techninės įrangos tiek fiziškai, tiek programiškai.

Duomenų saugyklos be patogios vartotojo sąsajos dažniausiai neturi didelės vertės. Reikalinga speciali sistema, kuri turi būti diegiama serveryje, kad būtų galima lengvai prieiti prie saugyklos duomenų. Kaip ir pačios infrastruktūros, taip ir programinės įrangos įsigijimo ir priežiūros kaina gali būti nemaža. Taip pat reikia įvertinti ir programinės įrangos pasitikėjimo faktorių. Renkantis programą, reikia gerai išanalizuoti visas siūlomas galimybės. Reikia įvertinti, ar įgyvendinti visi išsikelti saugumo reikalavimai. Dažnai didesnė kaina nebūtinai reiškia geresnį produktą.

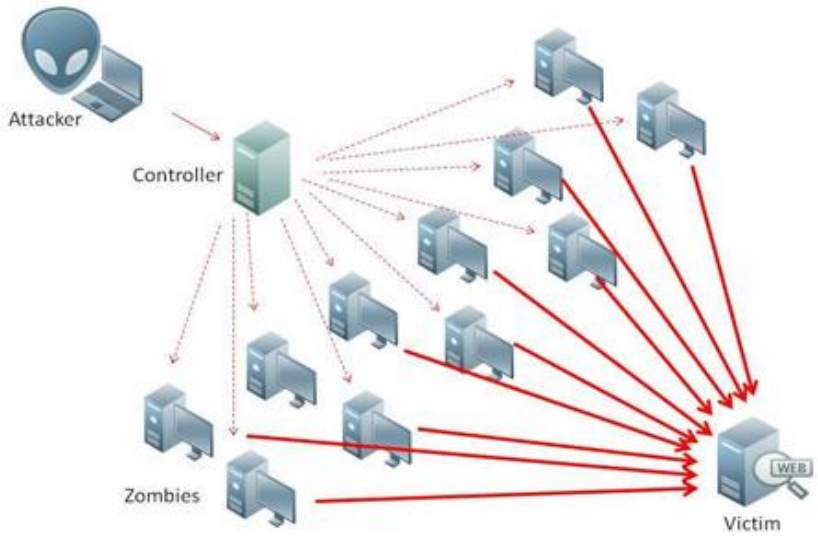
Sistemos praplėtimo problema atsiranda, kai nebeužtenka turimų resursų ir reikia galvoti apie saugų infrastuktūros gerinimą. Diegiant duomenų saugyklą, reikia gerai įvertinti ateities perspektyvas ir suprasi, kokie pokyčiai gali laukti ateityje. Dažnai geresnis sprendimas yra iš anksto numatyti, kokio tipo ir dydžio serverių reikės, bet paprastai tai padaryti yra labai sunku.

Duomenų sugadinimo problemos gali būti sprendžiamos įvairiais būdais. Ekspertai rekomenduoja nuolat daryti duomenų kopijas. Nesvarbu, ar tai būtų fizinis asmuo, ar įmonė. Tam reikia papildomų resursų. Tame pačiame diske ir kompiuteryje laikyti duomenų kopijas taip pat nėra saugu, taigi rekomenduojama turėti papildomus įrenginius. Geriausia kopijas laikyti net ne tame pačiame pastate, kuriame veikia originalios informacijos saugykla.

Saugumo klausimas išlieka bene pats svarbiausias iš visų. Visuomet reikia manyti, kad saugomus duomenis gali pasiekti bet kas, nesvarbu kokių priemonių imamasi. Saugumo spragos atrandamos netikėtai ir gali užtrukti, kol bus pastebėtos ir sutaisytos. Kenkėjai nuolat ieško kaip įsilaužti į sistemas ir pavogti duomenis. Kompanijose turi būti diegiama sugriežtina duomenų apsaugos politika ir turi būti prižiūrima, kad nuostatų būtų laikomasi nuolatos. Darbuotojai dažnai nepaiso saugumo reikalavimų ir nesivadovauja patarimais. Vienas nesaugus taškas sistemoje kelia grėsmę visai

sistemai. Silpni autentifikavimo sprendimai gali sukelti didelių problemų. Atakuotojai naudojasi tokiomis progomis ir spėlioja vartotojų slaptažodžius. Gali būti panaudota ir grubios jėgos ar žodyno ataka.

Įvairios atakos yra naudojamos įsibrauti į sistemas ar tiesiog jas nulaužti ir padaryti neprieinamas. SQL injekcijos atakos vis dar įmanomos šiais laikais, nes programuotojai neatsakingai priima sprendimus kurdami programinę įrangą. Atakuotojai nepaiso gresiančių nuobaudų ir visokiomis priemonėmis bando pakenkti sistemoms. Vis dar pasitaiko atvejų, kai yra naudojama paskirstyta paslaugos trikdymo ataka (1.1 pav.), kuomet vyksta intensyvus serverio apkrovimas dideliu kiekiu užklausų, dėl kurių tampa neįmanoma aptarnauti likusius klientus.



1.1 pav. DDoS atakos iliustracija (šaltinis: https://www.howtogeek.com/281707/what-are-denial-of-service-and-ddos-attacks/)

Virusai ir bet kokia kita kenkėjiška programinė įranga kelia didelę grėsmę net tik duomenų saugykloms, bet ir bet kokiai sistemai. Šnipinėjimo įrankiai gali sugauti jautrią informaciją ir padalinti ją trečiosioms šalims.

Pasitaiko atvejų, kai duomenų saugyklose naudojamos silpnos šifravimo schemos ir mechanizmai ar visai nenaudojamas joks panašus sprendimas. „Plikų“ duomenų laikymas prilygsta informacijos padėjimui ant lėkštutės, nes visuomet turime galvoti, kad bet kuriuo momentu galime prarasti duomenų kontrolę. Įsibrovėlis, gavęs užšifruotus duomenis galimai neturės galimybių juos perskaityti. Svarbu naudoti tinkamas šifravimo metodikas ir duomenis slėpti naudojant kuo stipresnius apsaugos sprendimo mechanizmus.

Ištrinti duomenys iš kietojo disko dažniausiai lieka kaupiklyje, tik nėra parodomas jų egzistavimas. Likę nematomi duomenų likučiai gali neautorizuotiems asmenims suteikti galimybę atsekti informaciją pagal likusius pėdsakus. Sistemas administruojantys asmenys turėtų pasirūpinti, kad ištrinti duomenys būtų sunaikinti iš tiesų.

Fizinė apsauga svarbi ne mažiau nei skaitmeninė. Dažnai neskiriama pakankama dėmesio pačių serverių apsaugai ir prieigos kontrolei. Jei bet kuris darbuotojas įmonėje gali fiziškai pasiekti mašiną su duomenimis, ji yra nesaugi. Šiame pasaulyje negalima pasitikėti niekuo. Išgauti duomenis iš nesaugomo serverio gali būti vieni juokai.

Problemoms spręsti pateikiami konkretūs pasiūlymai ir rekomendacijos įvairiuose šaltiniuose. Problemas bandoma spręsti tokiais būdais:

- diegiama sugriežtinta duomenų apsaugos politika;
- sugriežtinta prieigos kontrolė;
- stiprūs kriptografiniai ir šifravimo sprendimai;
- duomenų praradimo prevencija;
- stipri tinklo apsauga;
- atsarginių kopijų darymas ir tinkamas jų saugojimas [1].

1.4. Failų sistemų analizė

Failų sistema – tai sistema atsakinga už duomenų saugojimą fiziniame įrenginyje (pvz.: kietajame diske ar *flash* atminties laikmenoje). Failų sistemos veikia remiantis įvairiomis duomenų struktūromis, sąsajomis ir algoritmais. Vienos geriausiai žinomų ir plačiausiai naudojamų failų sistemų:

- *Unix* šeimos operacinės sistemos failų sistemos: EXT, EXT2, EXT3, EXT4, XFS, JFS, Btrfs (Linux), UFS (Solaris), APFS (macOS);
- *DOS* ir *Windows* operacinėse sistemose naudojamos failų sistemos: FAT (FAT-12, FAT-16, FAT-32), NTFS, exFAT.

Pastarosios sistemos yra vietinės failų sistemos. Tai reiškia, kad jos nėra paskirstytos ir veikia lokalioje mašinoje. Egzistuoja ir paskirstytosios (tinklinės) failų sistemos. Tai sistemos, kurios veikia nutolusiuose kompiuteriuose (serveriuose) saugant pilnas failų sistemos kopijas arba tam tikras dalis ir duomenų mainus atliekat per tinklo protokolus. Informacijos saugojimo sprendimai, pagrįsti paskirstyta failų sistema, auga vis sparčiau ir sparčiau. Taip yra todėl, kad vyksta staigus ir nuolatinis nestruktūrizuotų duomenų augimas įmonių duomenų centruose. Kiekvienais metais didėjant duomenų kiekiui, organizacijos stengiasi išnaudoti paskirstytas failų sistemas, kad būtų galima lengviau valdyti didesnius duomenų kiekius pasiekiant geresnį našumą ir išleidžiant mažiau pinigų. Populiariausi ir didžiausi tokių sistemų kūrėjai bei vystytojai: IBM, Red Hat, Dell EMC, Cloudian, MinIO, SUSE [5].

Vietinėse failų sistemose konfidencialumo siekiama pasitelkiant įvairias priemones. Svarbius duomenis galima šifruoti kriptografijos pagalba. Tokiu būdu, duomenys bus pasiekiami tik toms šalims, kurios reikiamą raktą duomenims iššifruoti. Taip pat svarbu tvarkyti prieigos leidimus prie saugomų failų. Įprastai, sistemoje gali veikti kelios naudotojų paskyros, skirtos skirtingiems naudotojams ar veikiančioms paslaugoms kompiuteryje. Reikia užtikrinti, kad failų sistemoje saugomus duomenis galėtų tvarkyti tik tos paskyros, kurioms tai yra numatyta. Pavyzdžiui, jeigu sistemoje sukurta paskyra duomenų bazės paslaugai, tai ši paskyra turėtų turėti prieigą tik prie jai skirtų bylų ir neturėtų galėti modifikuoti sisteminių ar kito sistemos naudojo failų. Kompiuterio valdymas yra plati tema, apimanti daug esminių saugumo praktikų. Apsaugant įrenginius taip pat yra apsaugomi juose esantys duomenys. Laikantis įprastų saugumo praktikų, pvz.: naudojant antivirusinę sistemą, disko duomenų šifravimą bei ugniasienes, sustiprinamas ir duomenų konfidencialumo apsaugos aspektas.

Vientisumui užtikrinti naudojamos kontrolinės sumos. Tai pagal tam tikrą algoritmą (maišos funkciją) suskaičiuojama reikšmė, pagal kurią galima matyti, ar duomenys nekito nuo jų sukūrimo pradžios. Sukūrus duomenis, apskaičiuojama kontrolinė suma. Vėliau tuos pačius duomenis galima tikrinti apskaičiuojant kontrolinę sumą dar kartą ir palyginant abi reikšmes. Jei buvo atliktas bent menkiausias pakeitimas, reikšmės bus skirtingos ir taip žinosime, kad turime vienaip ar kitaip paveiktą pradinių duomenų versiją. Kriptografinių maišos funkcijų naudojimas tapo vientisumo užtikrinimo standartu. Vienos populiariausių ir plačiausiai naudojamų funkcijų – MD5 [6] ir SHA1 [7].

1.4.1. Paskirstytosios (tinklinės) failų sistemos

Paskirstyta failų sistema yra sprendimas duomenų saugojimui ir prieigai pagal kliento-serverio architektūrą. Duomenys saugomi centriniame saugojimo įrenginyje, tačiau juos galima pasiekti ir tvarkyti taip, lyg jie būtų saugomi vietiniame kliento kompiuteryje. Naudodami paskirstytą failų sistemą, naudotojai gali keistis informacija lengvai kontroliuojamu būdu.

Paskirstyta failų sistema pasižymi tokiomis savybėmis:

- pagerintas failų pasiekiamumas, prieigos laikas ir tinklo efektyvumas;
- didesnis praplečiamumas (angl. scalability);
- prieigos prie duomenų skaidrumas ir vietos nepriklausomumas;
- veiksmingas apkrovos balansavimas.

Pasiskirstytos failų sistemos suteikia toleranciją gedimams ir aukštą suderinamumą kaupiant duomenis daugelyje mazgų. Rinkoje, kurioje gebėjimas maksimaliai padidinti duomenų vertę yra labai svarbus siekiant sėkmės versle, reikalingi duomenų saugojimo sprendimai, kurie galėtų pagerinti prieigą prie duomenų ir suteikti geresnes galimybes jų analizei. Didelis našumas tokioms sistemoms

dažniausiai nėra pagrindinis uždavinys, nes didžioji duomenų dalis paprastai skirta archyvavimui ir ilgalaikiam saugojimui. Todėl paprastai nėra poreikio tokius duomenis įrašyti ar atgauti greitai, svarbiausiai juos patikimai ir saugiai išsaugoti.

Vis dėlto, šiuolaikiniai darbo krūviai tai pakeitė ir šiandien nei viena įmonė nenori, kad paskirstyta failų sistema nesugebėtų greitai ir efektyviai aptarnauti verslo poreikių. Nebepakanka vien tik gebėjimo saugoti didelį kiekį informacijos – šiandien saugojimo sprendimai taip pat turi būti ir greiti, ir veiksmingi. Be to, aukštesnio lygio paskirstyta failų sistema turi teikti šiuos privalumus:

- mažesnės išlaidos, dėl kurių sumažėja energijos sąnaudos ir eksploatacijos išlaidos;
- geresnis išteklių panaudojimas, įgalinantis gauti didesnę grąžą iš investicijų į saugyklas;
- mažesnės bendros nuosavybės išlaidos.

Kaip ir bet kurioje kitoje informacijos saugojimo sistemoje, taip ir paskirstytoje, informacijos konfidencialumas, vientisumas ir pasiekiamumas išlieka svarbiausi uždaviniai.

Konfidencialumas reiškia prieigos leidimą tik įgaliotiems asmenims, uždraudžiant prieigą pašaliniams asmenims, kurie neturi teisės peržiūrėti apsaugoto turinio. Norint pasiekti konfidencialumo užtikrinimą, reikia atlikti tris veiksmus:

- pirmiausia turime patikrinti ir patvirtinti naudotojo tapatybę;
- tuomet galime patikrinti, ar naudotojas gali prieiti prie informacijos;
- turime apsaugoti informaciją nuo neteisėtų asmenų ir vykdytojų.

Populiariausi mechanizmai, naudojami kieno nors tapatybei įrodyti, yra pagrįsti autentifikavimo procesu. Autentifikavimo procesas turi būti tinkamai paruoštas darbui tinklo aplinkoje. Ypač tinkamas centralizuotas modelis „Kerberos“. Pagal šį protokolą, naudotojo įgaliojimai saugomi centrinėje saugykloje [8]. Tai patogus būdas, tačiau nesuteikia galimybės patikrinti prieigos teisių operacijų vykdymo lygmenyje. Šiai problemai spręsti, serveryje galima išnaudoti laikinąją atmintį, kurioje išsaugomi pagrindiniai ir naujausi prisijungimo duomenys, kurie yra naudojami tuo laiko momentu. Dažniausiai naudojamas būdas patikrinti, ar naudotojas, kuris buvo autentifikuotas, gali pasiekti išteklius, yra prieigos kontrolės sąrašų (angl. ACL – Access Control List) naudojimas. Šie sąrašai rūpinasi kiekvienu įmanomu prieigos prie išteklių metodo deriniu ir suteikia leidimus tokio pobūdžio užklausoms [9]. Tokie sąrašai greitai auga ir gali tapti per dideli, todėl vietoj jų naudojama rolėmis pagrįsta prieigos teisių kontrolė RBAC (angl. Role Based Access Control) [10]. Įsitikinę, kad naudotojo tapatybė yra žinoma, ir kad galime pasakyti, ar jis gali pasiekti prašomą informaciją, reikia užtikrinti, kad perdavimo metu informacija nebūtų prieinama pašaliniams sistemos naudotojams. Įprastu atveju, tai pasiekiama kriptografijos taikymo dėka. Tradiciškai kriptografija – tai mechanizmas, naudojamas apsaugoti saugomą ir tinklu siunčiamą informaciją. Kriptografinės operacijos reikalauja nemažų sistemos resursų skaičiavimams atlikti, todėl paprastai jos naudojamos tik labai jautriems duomenims. Tinklo ir paskirstytų failų sistemų kontekste, reikia apsvarstyti, kas turėtų užšifruoti duomenis: klientai prieš pat siunčiant duomenis per tinklą, ar serveris, kai tik juos gauna. Jei duomenys užšifruoti klientų, nereikėtų naudoti šifruoto ryšio, nes perduodami duomenys jau yra užšifruoti. Tai taip pat sumažintų našumo problemą ir užtikrintų geresnį failų sistemos išplečiamumą, nes sunkiausias skaičiavimo operacijas atliktų klientai. Tai įgalintų užšifruotą failų sistemą, nes failai būtų saugomi be iššifravimo galimybės serverio pusėje. Kriptografinių priemonių naudojimas paskirstytoms failų sistemoms suteikia konfidencialumą. Taip pat nustačius kelis saugumo lygius, brangiausias operacijas taikyti galima tik tiems failams, kuriems to iš tikrųjų reikia, ir kuriuos verta apsaugoti stipresniais metodais.

Vientisumo užtikrinimas suteikia galimybę nustatyti, ar trečiosios šalys nepakeitė perduodamos informacijos. Vientisumas užtikrinamas naudojant kriptografines maišos funkcijas ir skaitmeninį parašą. Kriptografinės maišos funkcijos apskaičiuoja fiksuoto ilgio reikšmę iš perduodamų duomenų. Priėmus duomenis, maišos reikšmė apskaičiuojama dar kartą ir taip patikrinama, ar duomenys nebuvo pakeisti perdavimo metu. Maišos funkcijos turi keletą savybių, dėl kurių jos yra labai naudingos:

- šiek tiek pakeitus pradinius duomenis, maišos reikšmė pasikeičia į visiškai naują ir nepanašią į pirminę;

- šios funkcijos yra labai greitai apskaičiuojamos, todėl gali būti naudojamos didelės apkrovos sistemose be didesnių veikimo nuostolių.

Piktavalis sistemos naudotojas gali pakeisti maišos reikšmes ir perduoti jas kartu su pakeistais duomenimis. Norint to išvengti, naudojamas skaitmeninis parašas. Skaitmeninis parašas yra susijęs su autentifikavimu ir konfidencialumu. Pilnas procesas sudaro duomenų maišos vertės apskaičiavimą ir šios vertės šifravimą tokiu būdu, kad būtų galima įsitikinti siuntėjo tapatybe. Problema, su kuria susiduriame yra ta, kad tokiu būdu atsiranda papildomos operacijos, dėl kurių nukenčia visos sistemos našumas. Paskirstytose failų sistemose vientisumą užtikrina tinklo kriptografiniai protokolai ir pati failų sistema, kuri fiziškai saugo failus. Svarbiausias klausimas paskirstytose sistemose užtikrinant duomenų vientisumą: ar tai turėtų būti daroma blokų, ar failų lygmenyje. Jei tai daroma failų lygyje, esame priversti perskaityti visą failą, kad galėtume su juo atlikti bet kokias operacijas. Jei failas padalintas per kelis saugojimo įrenginius, gali kilti papildomų problemų atliekant sinchronizaciją. Jei vientisumo siekiame bloko lygyje, turėsime patikrinti kiekvieno failo bloką, norėdami surasti sugadintus ar pakeistus failus failų sistemoje (atsižvelgdami į jo vientisumo maišos reikšmes). Sprendžiant, kurį iš šių variantų pasirinkti, reikia nustatyti prieigos prie duomenų metodą ir tikslą. Ideali sistema turėtų galimybę leisti naudoti abu variantus vienu metu arba leisti pasirinkti vieną pagal poreikį. Reikia išanalizuoti, ar pateiktas vientisumo lygis atitinka keliamus lūkesčius, ir pasirinkti tarp šifravimo, šifravimo ir maišos ar tiesiog maišos funkcijos.

Pasiekiamumas reiškia galimybę pasiekti reikiamą informaciją bet kuriuo momentu, kai to reikalaujama. Pasiekiamumas ir jo kokybė priklauso nuo aparatinės, programinės įrangos, operacinės sistemos ir kitų susijusių veiksnių. Tai nėra lengvai užtikrinama ir garantuojama. Vienos pagrindinių atakų, kurios trikdo sistemų prieinamumą – tai „DoS“ ir „DDoS“ atakos, kai atliekama daug užklausų į vieną serverį ir serveris nebesugeba aptarnauti eilinių sistemos naudotojų. Pasiekiamumui užtikrinti dažnai naudojami metodai remiasi apkrovą balansuojančiais serveriais ir srauto filtravimu. Šie metodai dažniausiai nėra pilnas sprendimas paskirstytų failų sistemų pasiekiamumui užtikrinti, nes tai kelia papildomų problemų, siekiant išlaikyti serverių nuoseklumą ir vientisumą tarpusavyje. Srauto filtravimas gali padėti išvengti „DoS“ ir „DDoS“ atakų, nes tinklo failų sistemos paprastai yra prieinamos žinomam adresų diapazonui. Bet koks srautas, einantis iš arba į numatytuosius adresus, esančius už šio diapazonas, nėra leistinas. Taip pat galima imtis papildomų priemonių, nutraukiant ryšį su klientais, kurie nesugebėjo prisijungti kelis kartus iš eilės. Kriptografija gali būti naudojama kaip autentifikavimo mechanizmas ir priemonė greitai nuspręsti, ar užklausą pateikia teisėtas operacijos vykdytojas. Kitas žingsnis duomenų vientisumo link – tikrosios duomenų struktūros slėpimas serverio saugojimo įrenginyje. Jei užpuolikas patektų į paskirstytosios failų sistemos saugyklą, jis negalėtų pasakyti, kuri informacija yra teisinga ir kaip ją išgauti. Tai nepadėtų tuo atveju, jei užpuolikas nuspręstų ištrinti visą serverio turinį, tačiau tai būtų naudinga tuo atveju, jei užpuolikas ieškotų tam tikrų failų. Metodai, kurie gali būti naudojami šiai idėjai realizuoti, gali būti tikrųjų vardų keitimas maišos reikšmėmis ir simbolinių nuorodų naudojimas, kad būtų išvengta visos katalogų struktūros pakartojimo kiekviename serveryje.

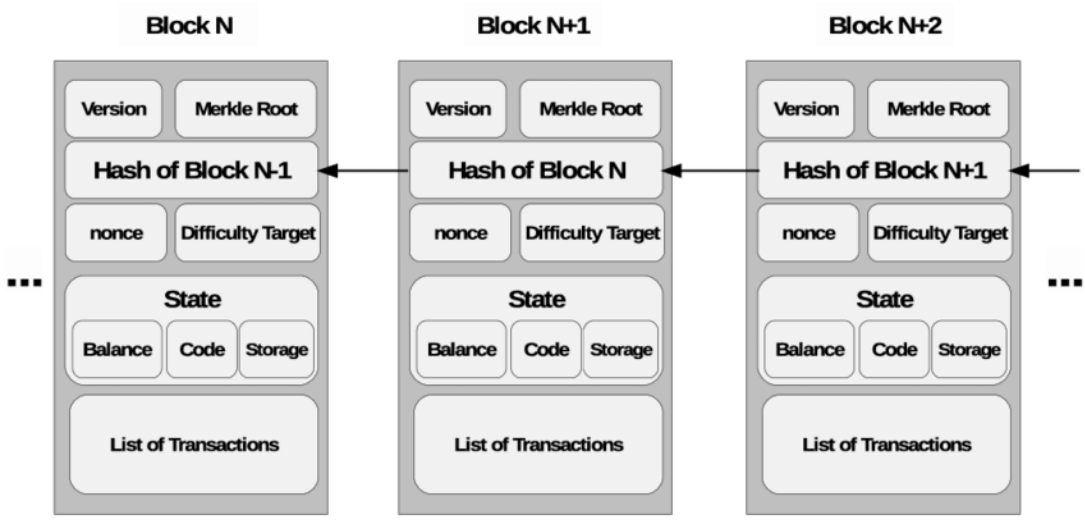
1.5. Blokų grandinės analizė, veikimo principai ir panaudojimas

Blokų grandinės technologija – tai blokų seka, sujungta į grandinę, kuri savyje saugo visą informacijos ir transakcijų judėjimo istoriją, o kartu ir aktualią informaciją (1.2 pav.). Kiekvienas blokas turi tėvinį bloką ir savyje saugo nuorodą į jį. Vienintelis blokas, kuris neturi tokios reikšmės – tai pradinis grandinės blokas, angliškai vadinamas *genesis block*. Blokų grandinę sudaro kelios pagrindinės dedamosios dalys: blokas, skaitmeninis parašas, konsensuso (ginčų sprendimo) algoritmai ir metodai bei įvairios savybės ir charakteristikos (1.2 pav.) [11]. Toliau ši technologija nagrinėjama detaliau.

Bloką sudaro antraštė ir pats pagrindinis informacijos kūnas. Antraštę sudaro:

- bloko versija – parodo, kokias bloko validavimo taisykles naudoti;
- *Merkle* šaknis – visų bloko transakcijų maišų reikšmių maišos reikšmė;
- šakninė maišos reikšmė (angl. hash) – tai apskaičiuota visų bloko transakcijų maišos reikšmė;

- laiko žyma – parodo, kada sukurtas blokas;
- *nBits* reikšmė – riba galiojančiai bloko maišos reikšmei;
- *nonce* reikšmė – tai keturių baitų ilgio laukas, kuris paprastai pradedamas skaičiuoti nuo 0 ir yra padidinamas su kiekvienu maišos reikšmių skaičiavimu;
- tėvinio bloko maišos reikšmė – tai 256 bitų ilgio reikšmė, kuri rodo į ankstesnį bloką (tėvinį). [11]



1.2 pav. Blokų grandinė ir blokai (šaltinis: https://www.researchgate.net/figure/Blockchain-design-structure-showing-chained-blocks-with-header-and-body-fields_fig2_321017113)

Bloko kūnas, ar kitaip sakant, pagrindinė informacinė dalis, sudaryta iš transakcijų skaitliuko ir pačių transakcijų. Maksimalus kiekis transakcijų, kurį gali blokas talpinti savyje, priklauso nuo paties pačio bloko bei transakcijų dydžio. Blokų grandinėse naudojami asimetriniai kriptografijos mechanizmai ir algoritmai, kurie yra reikalingi transakcijų validavimui ir tvirtinimui.

Skaitmeniniai parašai (paremti asimetrine kriptografija) naudojami nepatikimoje aplinkoje. Kiekvienas naudotojas privalo turėti porą raktų – privatųjį ir viešąjį. Privatusis raktas yra laikomas konfidencialiai ir nėra viešinamas. Toks raktas yra naudojamas transakcijoms pasirašyti. Pasirašytos transakcijos siunčiamos į visą blokų grandinės tinklą. Įprastą situaciją sudaro du etapai: pasirašymas ir patikrinimas. Tai galima nesunkiai iliustruoti. Šalis *A* pasirašo duomenis su savo privačiu raktu ir perduoda šaliai *B*. Taip įvykdomas pirmasis etapas – pasirašymas (kartu ir perdavimas). Tuomet, šalis *B* pasinaudodama viešuoju *A* raktu, patikrina, ar duomenys teisingi. Taip įvykdomas patikrinimo etapas. Tokiu būdu paaiškėja, ar duomenys nebuvo keisti perdavimo metu. Blokų grandinėse dažniausiai naudojamas ECDSA algoritmas (angl. Elliptic Curve Digital Signature Algorithm) (elipsinės kreivės skaitmeninio parašo algoritmas).

Toliau aptarsime pagrindines blokų grandinių charakteristikas:

- decentralizavimas. Tai visiška priešingybė įprastoms duomenų bazėms, kuriose kiekviena transakcija turi būti peržiūrima ir tvirtinama per centrinę administracinę sistemą. Tai kartais sukuria kamščius serveriuose ir operacijos užtrunka ilgiau nei tikimasi. Taigi, blokų grandinės sistemoje, viskas atvirkščiai – nėra jokio tarpininko, visas tinklas paskirstomas tarp naudotojų. Tokiu atveju, siekiant išlaikyti duomenų nuoseklumą paskirstytame tinkle, atsiranda poreikis spręsti konsensuso problemą pasitelkiant sutartas metodologijas ir įvairias technikas;
- patvarumas. Transakcijos patvirtinamos arba atmetamos pakankamai greitai (jei tinklas nedidelis) ir paprastai neįmanoma atstatyti arba atšaukti transakcijos, jei ji pakliūva į blokų grandinę. Neteisingos ir klaidingos transakcijos tinkle aptinkamos ir atmetamos gana greitai;
- anonimiškumas. Kiekvienas žmogus gali naudotis blokų grandinės sistema, neatskleisdamas savo tikrosios tapatybės. Naudotojas gauna identifikacinį numerį,

tačiau tai niekaip nėra susiję su realia žmogaus informacija. Jeigu pats neskelbia savo numerio viešai, atsekti žmogų gali būti itin sunku;

- aukštas audito galimybių lygmuo. Kiekviena transakcija turi turėti ryšį su praėjusiomis transakcijomis. Kai transakcija įrašoma į grandinę, ją lengva sekti ir atsekti. Tai leidžia peržvelgti visą įvykių istoriją, prieinamą viešai, dėl pačios blokų grandinės sistemos prigimties;

Blokų grandinės iš tikrųjų gali veikti kaip iš dalies centralizuotos sistemos. Iš viso galima išskirti tris blokų grandinių sistemų kategorijas:

- vieša blokų grandinė;
- privati blokų grandinė;
- konsorciumo blokų grandinė.

Vieša blokų grandinė, kaip jau sako pats pavadinimas, yra naudojama viešame tinkle ir pasiekiama visiems. Transakcijos tvirtinamos kiekviename mazge atskirai ir sprendimai atliekami bendrai visų tinklo mazgų mastu. Visi naudotojai turi prieigą prie visos saugomos informacijos ir gali laisvai matyti duomenų judėjimo istoriją. Transakcijos ir informacija apie jas, esančios blokuose, yra matomos beveik plika akimi visame tinkle. Turinį galima slėpti kriptografijos pagalba, priklausomai nuo kuriamos sistemos poreikio ir numatytų panaudos atvejų.

Konsorciumo tipo blokų grandinėje, duomenys juda tarp iš anksto numatytų mazgų ir bet kas negali naudotis tokia sistema. Čia šiek tiek pasireiškia centralizuotos sistemos principai, nes naudotojų ratas sudaro tam tikrą centralizuotą sistemą. Kiekvienas mazgas yra žinomas ir naujas į ratą priimamas tik gavus patvirtinimą iš visų šalių. Blokų informacija prieinama tik konsorciumui priklausantiems naudotojams.

Privati blokų grandinė labiausiai pasižymi centralizuotos sistemos savybėmis. Taip yra todėl, kad privačią blokų grandinę paprastai prižiūri viena organizacija. Duomenų mainai vyksta tik toje vienoje organizacijoje ir blokų grandinė nėra išleidžiama į išorę.

Tolimesnis trijų tipų palyginimas pateikiamas lentelėje žemiau (1.1 lentelė).

1.1 lentelė. Blokų grandinių tipų palyginimas [11]

Kriterijus	**Vieša blokų grandinė**	**Konsorciumo blokų grandinė**	**Privati blokų grandinė**
Konsensuso priėmimas	Dalyvauja kiekvienas mazgas	Pasirinkti mazgai	Viena organizacija
Skaitymo teisė	Vieša	Apribota, bet gali būti vieša	Apribota, bet gali būti vieša
Nekintamumas	Beveik neįmanomas	Įmanomas	Įmanomas
Efektyvumas	Mažas	Didelis	Didelis
Centralizavimas	Ne	Dalinis	Taip

Konsensuso sprendimas viešoje grandinėje atliekamas dalyvaujant visiems mazgams, kai konsorciumo grandinėje tai atlieka tik parinkti mazgai. Privačiojoje grandinėje situacija visai kitokia, nes galutinį sprendimą priima valdančioji organizacija.

Skaitymo teisės viešojoje sistemoje yra suteikiamos visiems, priešingai kituose tipuose, kur skaitymui gali būti suteikiama ir vieša prieiga, ir skirta tik uždaram naudotojų ratui.

Nekintamumas grandinėje beveik neįmanomas, jei kalbame apie viešąją blokų grandinę. Konsorciumo ir privačioje grandinėje tą pasiekti įmanoma, nes paprastai nėra tokio didelio naudotojų skaičiaus.

Efektyvumas priklauso nuo mazgų skaičiaus. Tikėtina, kad viešoje blokų grandinėje egzistuoja didelis kiekis naudotojų, taigi efektyvumas smarkiai krenta, nes kiekvienas mazgas turi patvirtinti kiekviena transakciją. Tuo tarpu privačioje ar konsorciumo grandinėje nėra tokio didelio kiekio naudotojų, taigi visos operacijos vykdomos daug sparčiau ir efektyviau.

Viešosios blokų grandinės atsiranda kiekvieną dieną ir pritraukia nemažai naudotojų. Geriausias panaudojimo pavyzdys – kriptovaliutos. Tuo tarpu privačiosios blokų grandinės gali būti naudojami įvairių organizacijų viduje, kaip saugi ir patikima duomenų bazė.

Konsensusui grandinėse naudojami įvairūs algoritmai. Keletas jų:

- PoW (angl. Proof of Work). Naudojamas *BitCoin* ir remiasi tuo, kad atsitiktinai parenkamas transakcijų prižiūrėtojas;
- PoS (angl. Proof of Stake). Tai PoW alternatyva su dėmesiu energijos taupymui;
- PBFT (angl. Practical Byzantine Fault Tolerance). Tai replikavimu paremtas algoritmas, skirtas toleruoti atsirandančias klaidas;
- DPOS (angl. Delegated Proof Of Stake). Remiasi tuo, transakcijų prižiūrėtojas parenkamas ne atsitiktinai.

Lentelėje pateikiamas jų palyginimas pagal mazgo tapatybės valdymą ir energijos sunaudojimą kartu su realiais pavyzdžiais (1.2 lentelė).

1.2 lentelė. Konsensuso algoritmų palyginimas [11]

Kriterijus	**PoW**	**PoS**	**PBFT**	**DPOS**
Mazgo tapatybės valdymas	Atviras	Atviras	Ribojamas	Atviras
Energijos sunaudojimas	Didelis	Vidutinis	Mažas	Vidutinis
Pavyzdys	*BitCoin*	*PeerCoin*	*Hyperledger Fabric*	*BitShares*

1.6. Blokų grandinių pranašumai saugiam informacijos saugojimui

Šiandien blokų grandinės technologija yra taikoma įvairiose srityse. Tai nestebina, nes ši technologija suteikia daug patogumų ir funkcijų. Galima išskirti keletą pagrindinių:

- plečiamumas ir išskaidomumas (angl. scaling);
- stebėjimas realiu laiku (angl. real-time monitoring);
- vartotojo izoliacija;
- momentinis turto ar informacijos perdavimas internetu;
- modulinis branduolys (angl. modular core) [12].

Atsižvelgiant į šiuos punktus, galime matyti, kad blokų branginės technologija gali būti naudojama informacijos vientisumui ir stebėjimui užtikrinti, o tai yra vienas svarbiausių aspektų kalbant apie informacijos apsaugą.

Tradiciniai duomenų apsaugos modeliai dažnai priklauso nuo nuolatinio pastiprinimo. Tai reiškia, kad jie nuolat tobulinami ir patobulinimai yra „klijuojami ant viršaus“ [13]. Pridedami papildomi sluoksniai autentifikavimui, prieigos kontrolei, saugesnis šifravimas ir kiti algoritmai. Tai galiausiai tampa sunku palaikyti ir suprasti, kaip viskas veikia ir kaip turėtų veikti iš tikrųjų.

Blokų grandinių technologija yra paremta decentralizavimu, taigi nėra vieno centrinio taško, nuo kuriuo priklausytų visa sistema. Tokią sistemą galima pavadinti skaitmeninių sandorių knyga ir kiekvienas kompiuteris, kuris naudojasi tokia sistema, turi nuosavą duomenų kopiją. Vieno valdančio vieneto nebūvimas suteikia sistemai daugiau saugumo ir skaidrumo. Vietoje to yra naudojami įvairūs susitarimo protokolai, naudojami visuose mazguose. Šių protokolų pagalba yra atliekamas transakcijų tvirtinimas ir duomenų įrašymas į grandinę. Nepaprastai svarbu, kad saugoma informacija būtų teisinga ir tiksli. Nustojus veikti keliems mazgams, informacija nedingsta ir tai yra didžiulis sistemų, paremtų blokų grandinėmis, pranašumas.

Blokų grandinių sistemas gana sunku nulaužti. Pavadinimas sako, kad tai grandinė skaitmeninių blokų, kuriuose saugoma įvykdytų transakcijų informacija. Kadangi jie nėra saugomi vienoje vietoje, o yra paskirsti tarp naudotojų (tarp kiekvieno mazgo), nėra vieno taško, kurį galimą būtų sugadinti ar kaip nors kitaip modifikuoti. Tai nuolat atsinaujinantis ir paskirstytas informacijos tinklas, kurio sinchronizacija palaikoma kiekvieną akimirką. Visi blokai susiję ir bandymai atliki pakeitimą iš karto griauna visą ryšį tarp jų, kas indikuoja nekorektišką sistemos būseną. Atakuotojai įsilaužę į įprastą centralizuotą duomenų bazę ar kitą saugyklą, gali duomenis nutekinti, gadinti ir keisti, o blokų grandinės sistemoje tokius veiksmus atlikti yra nepaprastai sunku ir beveik neįmanoma.

Galima manyti, kad blokų grandinių technologija yra puiki iš prigimties. Auga populiarumas ir naudojimo atvejų aibė. Kompanijoms tai atveria didžiules galimybes, siekiant sukurti saugią duomenų saugyklą. Tai ne vien saugi ir patikima sistema, o kartu ir skaidri, kur visą duomenų judėjimą galimą stebėti realiu laiku.

Blokų grandinių technologiją galima išskirti kaip atskirą duomenų bazių tipą [14]. Tai duomenų bazė, kuria galima tiesiogiai dalintis tarp galimai tarpusavyje nepasitikinčių šalių ir viena geriausių dalių – nereikalingas administratorius ar papildoma stebėjimo bei priežiūros sistema. Galima įžvelgti kontrastą tarp tradicinių reliacinių duomenų bazių (SQL), o taip pat ir tarp dokumentų tipo duomenų saugojimo duomenų bazių (NoSQL), kurios yra prižiūrimos ir palaikomos tam tikros dedikuotos šalies. Net jei ir naudojamas tam tikro lygio paskirstymas tradicinėse duomenų bazėse, tai nesuteikia visiško decentralizavimo ir veikimo principas išlieka visiškai kitoks lyginant su blokų grandinių tipo duomenų baze. Tačiau negalima sakyti, vienas ar kitas tipas yra geresnis už kitą. Tiek įprastos duomenų bazės, tiek blokų grandinės turi tam tikrų pranašumų ir trūkumų tarpusavyje. Palyginimą galima atlikti išskiriant kelis aspektus įvertinimui: konfidencialumas, tvirtumas, veikimo patikimumas.

Konfidencialumo labiau reikia tikėtis iš standartinių duomenų bazių. Kiekvienas mazgas blokų grandinėje atlieka patvirtinimą ir apdoroja transakciją. Tai yra įmanoma, nes mazgas turi pilną prieigą prie esamos duomenų bazės būsenos, prašomų transakcijos pakeitimų ir skaitmeninio parašo, kuris įrodo ir patvirtina transakcijos kilmę. Tai yra nepaprastai gudrus veikimo būdas, tačiau kaip jau galima pastebėti, nėra jokio konfidencialumo požymio - viskas skaidru ir matoma. Tai ypatingai netinkamas metodas finansinėms sistemos bei bet kokiai kitais sistemai, kurioje visi apie visus negali žinoti kiekvienos detalės ir turi būti užtikrinamas konfidencialumas. Įprastos duomenų bazės šios problemos išvengia apribojant transakcijų vykdymą tik tam tikriems naudotojams. Šie apribojimai įdiegti vienoje centrinėje vietoje. Dėl to, pilną duomenų komplektą galima matyti tik viename mazge, vienoje vienintelėje vietoje, priešingai blokų grandinei, kur informacija dalinamasi visame tinkle. Centralizuotose sistemose kontroliuojamas tiek duomenų nuskaitymas, tiek modifikavimas, kai tuo tarpu blokų grandinėje, kontroliuojamas tik naujų duomenų įrašymas. Žinoma, yra sprendimų ir blokų grandinėse, kurie taikomi pasiekti tam tikrą konfidencialumo lygį. Gali būti taikomos įvairios kriptografinės priemonės siekiant įgyvendinti konfidencialias transakcijas. Tai atsiliepia veikimo našumui, nes kuo daugiau skaičiavimų taikoma kiekvienos transakcijos vykdymo metu, tuo lėčiau veikia visa sistema. Kad ir koks būdas būtų taikomas informacijai slėpti blokų grandinėje, niekas nepralenks paprasčiausio visiško duomenų paslėpimo ir prieigos apribojimo tradicinėje duomenų bazės sistemoje.

Palyginime pagal tvirtumą, viršų ima blokų grandinėmis paremtos duomenų bazės. Šios sistemos pasižymi itin didele tolerancija klaidoms ir duomenų dubliavimui. Kiekvienas mazgas apdoroja kiekvieną transakciją, o tai reiškia, kad nei vienas mazgas nėra būtinas visos sistemos veikimui. Nustojus veikti vienam mazgui, toliau lieka veikti visas likęs tinklas. Kadangi kiekviena tinklo dalis susijungia tarpusavyje, gaunamas gan „tirštas“ tinklas, taigi nutrūkstantis ryšys tarp mazgų nekelia didelių problemų. Blokų grandinė turi savybę kiekvienam neveikimo laikotarpiu atsilikusiam mazgui leisti pasivyti dabartinę tinklo būseną. Įprastos duomenų bazės siūlo įvairius būdus ir technikas duomenų replikavimui, tačiau šiuo klausimu su blokų grandinėmis lygintis sunku, nes pagal dizainą – tai visai kitas lygmuo. Nereikalinga jokia papildoma konfigūracija, užtenka keliems mazgams susijungti ir automatiškai yra atliekamas informacijos sinchronizavimas tarpusavyje. Negana to, mazgai tinkle gali būti laisvai pridėti arba išimti, nesukeliant jokių problemų ir nereikalaujant jokio išankstinio perspėjimo kitiems dalyviams. Dar vienas privalumas – transakcijos gali būti išsiunčiamos tik vienam ar keliems mazgams ir padalijimas per visą tinklą atliekamas automatiškai. Tradicinės duomenų bazės reikalauja brangios infrastruktūros ir sudėtingų duomenų atkūrimo technikų nelaimės atveju, jeigu tokia sistemą norime laikyti tvirta. Brangi infrastruktūra taip pat reikalinga užtikrinant ir gerą bazės pasiekiamumą bet kuriuo momentu. Nuolat turi būti atliekamos duomenų kopijos. Idealu, jei tam naudojamos papildomos mašinos. Taip pat turėtų būti naudojamos atsarginės mašinos, skirtos perjungimui, jeigu nustotų veikti pagrindinis serveris. Visa tai kainuoja nemažus pinigus. Tiek palaikymas, tiek aparatinė įranga nėra pigus malonumas. Įsivaizduokime, kad veikia keli blokų grandinės sistemos mazgai visame pasaulyje paprastų žmonių kompiuteriuose. Kiekvienas mazgas

jungiasi tarpusavyje ir nesvarbu, jei keli komunikavimo keliai nustoja veikti. Nesvarbu, jei keli mazgai visai sugenda, nes tuo pačiu metu veikia dar kelios kopijos. Taip veikia beveik nenutrūkstantis tinklas. Tokiu būdu gauname gana pigią, tvirtą duomenų saugojimo sistemą.

Vertinant veikimo našumą, blokų grandinių sistemos, deja, nelaimi ir jos visuomet išliks lėtesnės už centralizuotas duomenų bazes. To priežastis nėra neištobulinta technologija, tiesiog iš prigimties ir dizaino pusės blokų grandinės negali varžytis spartos ir greičio klausimu. Apdorojant transakcijas blokų grandinė turi atlikti tas pačias operacijas, kaip ir tradicinė duomenų bazė, tačiau kartu egzistuoja ir papildomi faktoriai. Vienas iš jų – transakcijos parašo tikrinimas ir patvirtinimas. Kiekviena blokų grandinės transakcija turi būti pasirašoma skaitmeniniu būdu naudojant kriptografinius įrankius, pavyzdžiui, ECDSA algoritmą (angl. Elliptic Curve Digital Signature Algorithm). Tai būtina dėl transakcijos keliavimo tarp tinklo mazgų. Kitu būdu negalima patvirtinti šaltinio ir operacijos teisingumo. Tokių parašų generavimas ir apdorojimas yra pakankamai sunki operacija ir reikalauja sąlyginai nemažai laiko apskaičiavimui. Dėl to prarandama ypatingai daug laiko, jei tinklas yra plačiai naudojamas. Centralizuotoje duomenų bazėje užtenka kartą jungiantis įrodyti kas esame ir kitų operacijų metu to kartoti nereikia, taigi sutaupoma daug laiko. Dar viena iš priežasčių – konsensuso (ginčo sprendimo) mechanizmai. Paskirstytose bazėse reikia daugiau dėmesio skirti ginčų sprendimui tarp mazgų, nes kiekvienas veikia individualiai ir turi būti priimtas vieningas mechanizmas tokioms situacijoms spręsti. Operacijų metu vyksta intensyvus komunikavimo procesas tarp mazgų tinkle. Sprendžiamos tinklo išsišakojimo problemos ir panašūs veiksniai. Centralizuotose sistemose taip pat vyksta panašūs veiksmai, tačiau dėl operacijų atlikimo vienoje vietoje, nėra reikalingas toks didelis resursų kiekis ir tvarką palaikyti yra daug paprasčiau. Dar viena problema kyla ir dėl duomenų dubliavimo. Čia kalba neina apie individualius mazgus, o labiau susitelkiama į viso tinklo resursų kiekio poreikį. Centralizuotoje sistemoje transakcija apdorojama vieną kartą ir tuo viskas baigiasi, o blokų grandinėje ši operacija turi būti visame tinkle, t.y. kiekvienas mazgas atlieka tą patį veiksmą, taigi daug daugiau darbo reikia atlikti, siekiant to paties rezultato.

Blokų grandinių ir tradicinių duomenų bazių palyginimas koncentruotai pateikiamas lentelėje žemiau (1.3 lentelė).

1.3 lentelė. Blokų grandinių ir tradicinių duomenų bazių palyginimas [15]

Blokų grandinė	Tradicinė duomenų bazė
Nėra vieno prižiūrėtojo ar administratoriaus viešojoje blokų grandinėje	Yra administratorius ir centralizuotas operacijų kontroliavimas
Visi gali naudotis viešąja blokų grandine	Gali naudotis tik naudotojai su tam tikromis teisėmis
Visi gali įrašyti informaciją	Informaciją įrašyti gali tik naudotojai su tam tikromis teisėmis
Lėtas veikimas	Greitas veikimas
Nesunku peržiūrėti istorinius duomenis	Nėra istorinių duomenų, jei neįdiegtas papildomas funkcionalumas

1.7. Analizės išvados

Analizės metu apžvelgti informacijos saugojimo metodai ir atliktas jų palyginimas. Tradiciniai duomenų saugojimo būdai nėra blogas pasirinkimas kaupti informaciją šiomis dienomis. Technologijos nestovi vietoje ir nuolat atsiranda nauji metodai ir technologiniai sprendimai, taigi verta apžvelgti ir įvertinti, ką iš naujai siūlomų sprendimų galima bandyti taikyti nieko nelaukiant. Viena iš pakankamai naujų technologijų informacijai dalintis ir saugoti – blokų grandinės. Analizės skyriuje apžvelgtos šios technologijos charakteristikos ir atliktas palyginimas su tradiciniai informacijos saugojimo būdais. Pateikti pranašumai ir trūkumai lyginant su įprastomis centralizuotomis duomenų bazėmis.

Blokų grandinių technologija demonstruoja savo potencialą pasitelkdama pagrindines savo savybes: decentralizavimą, anonimiškumą, skaidrumą, tvirtumą. Išanalizavę šią technologiją, galime pastebėti ne vieną pranašumą prieš tradicinius sprendimus:

- sutrikdžius vieną mazgą (ar nustojus jam veikti), lieka daug grandinės kopijų visame tinkle, taigi informacija neprarandama ir ją lengva parsiųsti iš naujo;
- sunkus (praktiškai neįmanomas) duomenų klastojimas, nes transakcijos ir blokai validuojami daugelio mazgų pagalba visame tinkle;
- validacija daugelyje mazgų užtikrina sistemos ir duomenų vientisumą;
- į grandinę įrašytų duomenų pakeisti negalima – tai užtikrina duomenų vientisumą;
- tai decentralizuota (paskirstyta) sistema, taigi užtikrinamas pasiekiamumas.

Kaip ir bet kokia kita technologija, blokų grandinės nėra tobula technologija. Kuriant sistemą ir renkantis būdą duomenims saugoti, mūsų pasirinkimas greičiausiai priklausys nuo tam tikrų aspektų. Jei siekiame maksimalaus privatumo ir konfidencialumo, greičiausiai blokų grandinėmis grįstos saugyklos nesirinksime ir pasikliausime įprastais centralizuotais sprendimais. Konfidencialumą užtikrinti galima taikant papildomas kriptografines priemones, tačiau tai sumažina sistemos greitaveiką.

Blokų grandinės nėra skirtos didelės apimties duomenims saugoti. Realizuojant saugaus informacijos saugojimo metodą paremtą blokų grandinėmis, reikia turėti omenyje tai, kad norint saugoti didelius failus, privaloma naudoti dedikuotą išorinę saugyklą.

2. BLOKŲ GRANDINE PAREMTOS FAILŲ SISTEMOS PROJEKTAS

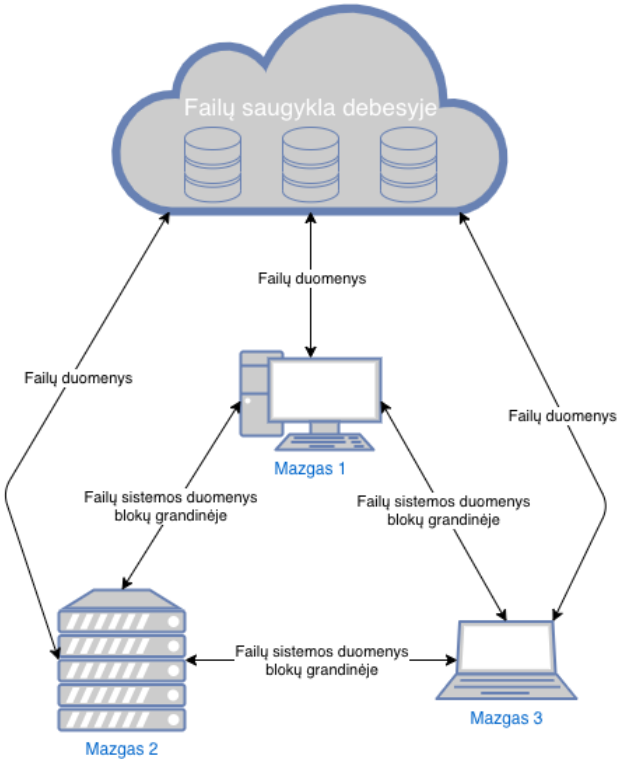
Atlikus esamų saugaus informacijos saugojimo metodų analizę ir pateikus išvadas, toliau aprašomas projektas naujam failų sistemos modeliui, kuris yra paremtas blokų grandinių technologija. Ši failų sistema bus skirta įvairaus dydžio statinėms byloms saugoti bei dalintis.

Sistemos projektas išskirstytas į šias dalis:

- infrastruktūros koncepcija;
- reikalavimų specifikacija (funkciniai ir nefunkciniai reikalavimai);
- blokų grandinės taikymas;
- failų sistemos operacijos;
- statinis sistemos vaizdas;
- dinaminis sistemos vaizdas;
- duomenų saugumo sprendimas;
- išvados.

Sistemą sudarys blokų grandinės mazgai, susijungę į bendrą tinklą. Mazgas gali būti bet kuris asmeninis kompiuteris ar serveris. Orientuojamasi į viešąją blokų grandinę, tačiau sistemos taikymas priklauso nuo konkrečios implementacijos ir gali būti taikoma privati blokų grandinė. Naudojant viešąją blokų grandinę bus reikalingos papildomos priemonės saugomo turinio apsaugai, nes ne visa informacija turi būti viešai prieinama ir skaitoma. Minimaliam veikimui reikalingas vienas pradinis mazgas (jų gali būti ir daugiau), apie kurį turės žinoti kiti tinklo nariai, norėdami užmegzti ryšį ir pradėti darbą su failų sistema. Esant daugiau naudotojų tinkle, pagrindinis mazgas dalinsis informacija apie kitus jau žinomus ir veikiančius mazgus, taigi jam vienam nereikės rūpintis naujų klientų aprūpinimu ir ryšio palaikymu. Tinklo apkrova ir mazgų susiejimas tarpusavyje turi vykti automatinio balansavimo principu apribojant maksimalų kiekį mazgų, su kuriuo gali jungtis vienas mazgas.

Blokų grandinė nėra skirta saugoti didelės apimties failams, todėl šią technologiją galime panaudoti tik failų sistemos branduoliui realizuoti, t.y. failų metaduomenų bei failų struktūros saugojimui. Patys failai turi būti saugomi atskiroje saugykloje debesyje. Kiekvienas sistemos mazgas atsakingas už failų sistemos teisingumą ir konkrečių duomenų paruošimą saugojimui išorinėje saugykloje. Koncepcinis infrastruktūros modelis (2.1 pav.) vaizduoja, kaip tokia sistema atrodo iš aukšto lygmens.



2.1 pav. Koncepcinis infrastruktūros modelis

Čia vaizduojama sistema, kurioje veikia trys mazgai skirtingo tipo mašinose. Bet kuriuo laiko momentu prie tinklo gali prisijungti daugiau kompiuterių. Failų sistemos mazgas gali veikti dedikuotame serveryje ar bet kuriame asmeniniame kompiuteryje. Kiekvienas tinklo dalyvis gali pasiekti duomenų saugyklą ir parsiųsti prieinamus failus, o failų sistemos struktūros teisingumas ir duomenų integralumas palaikomas blokų grandinėje, už kurios prižiūrėjimą yra atsakingas kiekvienas prisijungęs mazgas.

2.1. Reikalavimų specifikacija

Projektui apibrėžti kelių tipų reikalavimai:

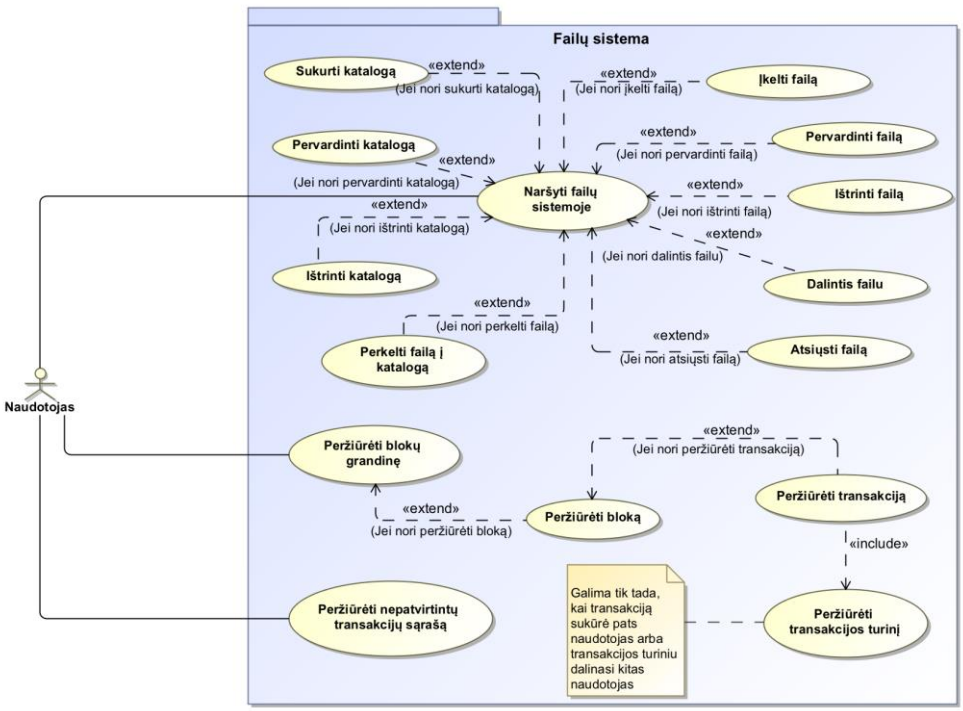
- funkciniai reikalavimai;
- nefunkciniai reikalavimai;
- reikalavimai naudotojo sąsajai.

Funkciniai reikalavimai pavaizduoti panaudos atvejų diagrama, kuri yra suskirstyta į du paketus: *Failų sistema* ir *blokų grandinės mazgas*. Kiekvienas panaudos atvejis detaliai aprašytas 2.1 - 2.30 lentelėse.

Nefunkciniai ir naudotojo sąsajos reikalavimai pateikiami glaustai bendram vaizdui sudaryti (rekomendacinio pobūdžio), nes tai nėra esminė siūlomo metodo dalis.

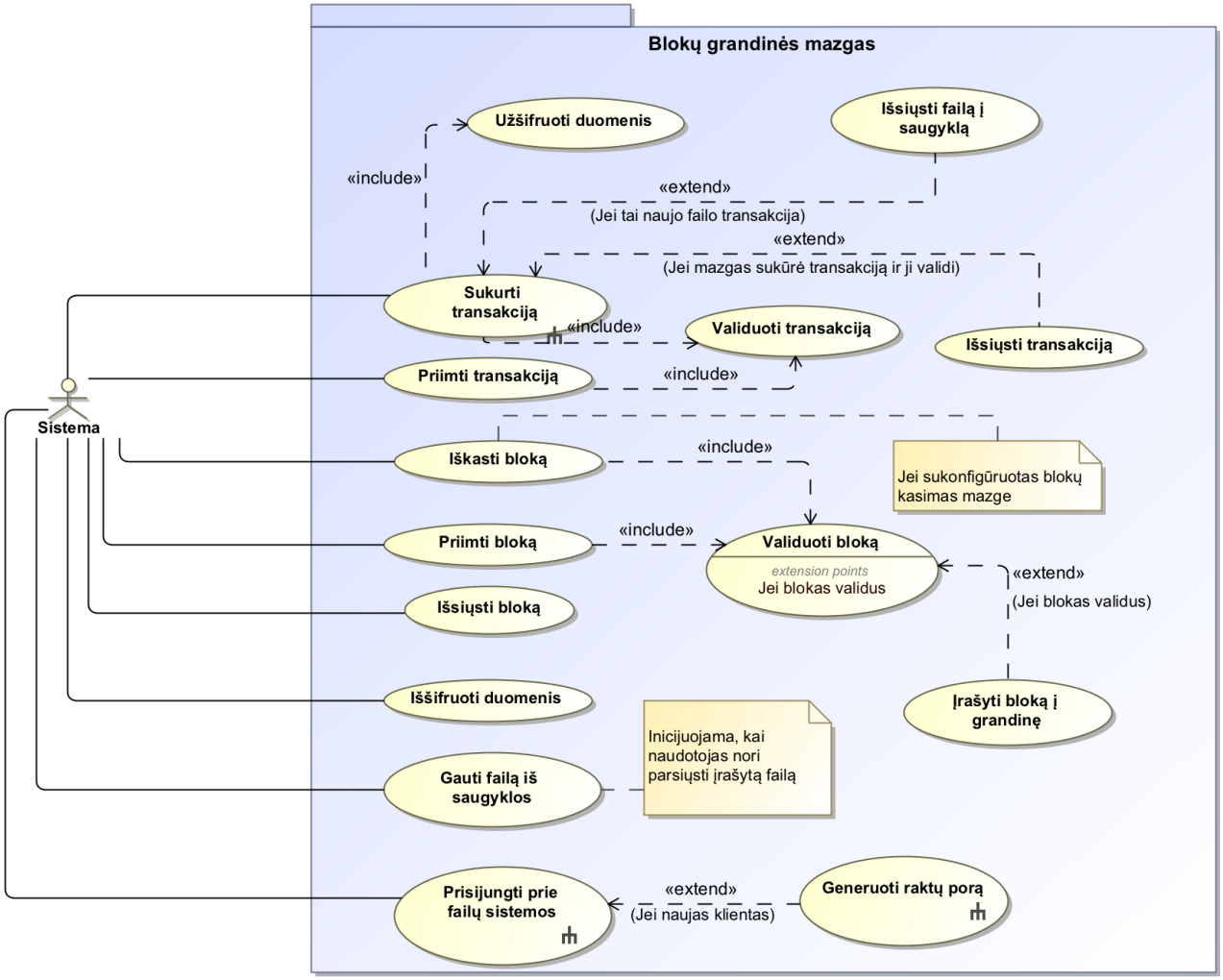
2.1.1. Panaudos atvejai (funkciniai reikalavimai)

Funkciniai reikalavimai aprašo pagrindines sistemos funkcijas. Pakete *Failų sistema* pateikti panaudos atvejai, kurie yra tiesiogiai susiję su naudotoju ir jo veiksmais (2.2 pav.). Šiuos panaudos atvejus sudaro tokie veiksmai: naršyti failų sistemoje, įkelti failą, pervardinti failą, ištrinti failą, dalintis failu, atsiųsti failą, sukurti katalogą, pervardinti katalogą, ištrinti katalogą, perkelti failą į katalogą, peržiūrėti blokų grandinę, peržiūrėti bloką, peržiūrėti transakciją, peržiūrėti transakcijos turinį ir peržiūrėti nepatvirtintų transakcijų sąrašą. Didžioji dalis veiksmų siejasi su failų sistemos operacijomis. Kita dalis veiksmų yra tiesiogiai susijusi su blokų grandine ir jos peržiūra. Kadangi blokų grandinė yra viešai prieinama visiems sistemos naudotojams, kiekvienas turi galimybę ją peržiūrėti. Pagrindinė bloko dedamoji dalis transakcija, taigi jas taip pat galima peržiūrėti naršant blokų grandinėje, bet ne visų transakcijų turinį naudotojas gali matyti. Detali transakcijos peržiūra galima tik tada, jei ją sukūrė pats naudotojas arba kažkas kitas suteikė teisę į peržiūrą, pvz.: kitas naudotojas pasidalino failu.



2.2 pav. Failų sistemos panaudos atvejai

Panaudos atvejai, susiję su techninėmis sistemos funkcijomis, atskirti į paketą *blokų grandinės mazgas* (2.3 pav.). Tai blokų grandinės veikimą, failų apdorojimą, mazgų bendravimą užtikrinančios operacijos, kurias atlieka sistema. Šiuos panaudos atvejus sudaro veiksmai: prisijungti prie failų sistemos, generuoti raktų porą, sukurti transakciją, užšifruoti duomenis, validuoti transakciją, išsiųsti failą į saugyklą, išsiųsti transakciją, priimti transakciją, priimti bloką, iškasti bloką, validuoti bloką, įrašyti bloką į grandinę, išsiųsti bloką, iššifruoti duomenis, gauti failą iš saugyklos.



2.3 pav. Blokų grandinės mazgo panaudos atvejai

Kiekvieno panaudos atvejo struktūrizuotas aprašymas pateikiamas lentelėse žemiau (2.1 - 2.30 lentelės).

2.1 lentelė. PA „Naršyti failų sistemoje“ aprašymas

PA „Naršyti failų sistemoje“		
Paketas: Failų sistema		
Tikslas: Peržiūrėti įkeltus failus, sukurtus katalogus bei atlikti standartines failų sistemos funkcijas		
Aprašymas: Naudotojas gali naršyti failų sistemoje, peržiūrėti savo sukurtus katalogus ir įkeltus dokumentus		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Naudotojo sąsajoje pasirenkamas failų peržiūros langas
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	Įkelti failą Pervardinti failą Ištrinti failą Dalintis failu

		Atsiųsti failą Sukurti katalogą Pervardinti katalogą Ištrinti katalogą Perkelti failą į katalogą
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Meniu parenkamas failų peržiūros punktas		1. Sistema užkrauna šakninį (arba paskutinį peržiūrėtą) katalogą ir atveria failų naršymo langą 1.1. Pateikiami komponentai išplečiančių PA vykdymui
Po sąlygos:		Atvertas failų naršymo langas

2.2 lentelė. PA „Įkelti failą“ aprašymas

PA „Įkelti failą“		
Paketas: Failų sistema		
Tikslas: Išsaugoti failą failų sistemoje		
Aprašymas: Naudotojas gali pasirinkti failą iš savo kompiuterio ir įkelti jį į pasirinktą failų sistemos katalogą		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Failų naršymo lange paspaudžiamas naujo failo įkėlimo mygtukas
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Spaudžiamas naujo failo įkėlimo mygtukas 2. Pasirenkami failai iš kompiuterio atminties 3. Spaudžiamas patvirtinimo mygtukas		1. Atveriamas kompiuterio failų naršymo ir parinkimo langas 2. Pasirinkti failai vaizduojami atskiroje sekcijoje 3. Pasirinkti failai paruošiami įkėlimui, sukuriama transakcija 3.1. Naudotojas grąžinamas į sistemos langą
Po sąlygos:		Pasirinkti failai išsaugomi failų sistemoje
Alternatyvūs scenarijai		
1. Kataloge jau egzistuoja failas su tokiu pačiu pavadinimu		1. Failas neįkeliamas 1.1. Parodomas informacinis pranešimas

2.3 lentelė. PA „Pervardinti failą“ aprašymas

PA „Pervardinti failą“		
Paketas: Failų sistema		
Tikslas: Pervardinti jau įkeltą failą		
Aprašymas: Naudotojas gali pakeisti pavadinimą bet kuriam jau įkeltam failui		
Prieš sąlyga:		Paleistas sistemos mazgas Įkeltas bent vienas failas
Sužadinimo sąlyga:		Pasirinktam failui parenkama failo pervardinimo funkcija
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**

1. Pasirenkamas failas 2. Pasirenkama failo pervardinimo funkcija 3. Įrašomas naujas pavadinimas 4. Patvirtinamas naujas vardas	1. Atidaromos failo funkcijos 2. Atveriamas langas pavadinimo keitimui 3. Atvaizduojamas pakeitimas 4. Sukuriama nauja transakcija
Po sąlygos:	Pakeistas failo pavadinimas
Alternatyvūs scenarijai	
1. Kataloge jau egzistuoja failas nauju vardu	1. Operacija atmetama 1.1. Parodomas informacinis pranešimas

2.4 lentelė. PA „Ištrinti failą“ aprašymas

PA „Ištrinti failą“		
Paketas: Failų sistema		
Tikslas: Pažymėti failą kaip ištrintą ir pašalinti iš failų saugyklos		
Aprašymas: Naudotojas gali pašalinti failą iš failų sistemos		
Prieš sąlyga:		Paleistas sistemos mazgas Įkeltas bent vienas failas
Sužadinimo sąlyga:		Pasirinktam failui parenkama failo ištrynimo funkcija
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Pasirenkamas failas 2. Pasirenkama failo trynimo funkcija		1. Atidaromos failo funkcijos 2. Sukuriama nauja failo ištrynimo transakcija
Po sąlygos:		Failas neberodomas failų sistemoje ir yra pašalintas iš failų saugyklos

2.5 lentelė. PA „Dalintis failu“ aprašymas

PA „Dalintis failu“		
Paketas: Failų sistema		
Tikslas: Pasidalinti failu su kitu naudotoju		
Aprašymas: Naudotojas gali leisti parsiųsti savo įkeltą failą kitam failų sistemos naudotojui		
Prieš sąlyga:		Paleistas sistemos mazgas Įkeltas bent vienas failas Žinomas kito naudotojo kodas
Sužadinimo sąlyga:		Pasirinktam failui parenkama failo dalinimosi funkcija
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Pasirenkamas failas 2. Pasirenkama failo dalinimosi funkcija 3. Įrašomas kito naudotojo kodas 4. Atliekamas patvirtinimas		1. Atidaromos failo funkcijos 2. Atveriamas langas naudotojo kodo įrašymui 3. Atvaizduojamas naudotojo kodas 4. Sukuriama nauja failo dalinimosi transakcija
Po sąlygos:		Kitas naudotojas mato failą ir gali jį parsiųsti

2.6 lentelė. PA „Atsiųsti failą“ aprašymas

PA „Atsiųsti failą“		
Paketas: Failų sistema		
Tikslas: Atsiųsti failą iš failų sistemos		
Aprašymas: Naudotojas gali atsiųsti failą ir išsaugoti jį savo kompiuteryje		
Prieš sąlyga:		Paleistas sistemos mazgas Įkeltas bent vienas failas
Sužadinimo sąlyga:		Pasirinktam failui parenkama failo atsiuntimo funkcija
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	Gauti failą iš saugyklos
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Pasirenkamas failas 2. Pasirenkama failo atsiuntimo funkcija		1. Atidaromos failo funkcijos 2. Atsiunčiamas failas į naudotojo kompiuterį 2.1. Kompiuteryje sukuriama identiška katalogų struktūra 2.2. Parsiunčiamas failas iš saugyklos 2.3. Failas išsaugomas naudotojo kompiuteryje
Po sąlygos:		Failas atsiųstas ir išsaugotas naudotojo kompiuteryje

2.7 lentelė. PA „Sukurti katalogą“ aprašymas

PA „Sukurti katalogą“		
Paketas: Failų sistema		
Tikslas: Sukurti naują katalogą failų sistemoje		
Aprašymas: Naudotojas gali sukurti naują katalogą pasirinktu pavadinimu		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Pasirenkama funkcija katalogo kūrimui
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Pasirenkama funkcija kurti naują katalogą 2. Įvedamas katalogo pavadinimas 3. Operacija patvirtinama		1. Parodomas naujo katalogo kūrimo langas 2. Atvaizduojamas naujo katalogo pavadinimas 3. Sukuriama nauja transakcija
Po sąlygos:		Sukurtas naujas katalogas
Alternatyvūs scenarijai		
1. Katalogas kuriamu pavadinimu jau egzistuoja		1. Operacija atmetama 1.1. Parodomas informacinis pranešimas

2.8 lentelė. PA „Pervardinti katalogą“ aprašymas

PA „Pervardinti katalogą“		
Paketas: Failų sistema		
Tikslas: Pervardinti jau sukurtą katalogą		
Aprašymas: Naudotojas gali pakeisti pavadinimą bet kuriam jau sukurtam katalogui		
Prieš sąlyga:		Paleistas sistemos mazgas Sukurtas bent vienas katalogas
Sužadinimo sąlyga:		Pasirinktam katalogui parenkama pervardinimo funkcija

Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Pasirenkamas katalogas 2. Pasirenkama katalogo pervardinimo funkcija 3. Įrašomas naujas pavadinimas 4. Patvirtinamas naujas vardas		1. Atidaromos katalogo funkcijos 2. Atveriamas langas pavadinimo keitimui 3. Atvaizduojamas pakeitimas 4. Sukuriama nauja transakcija
Po sąlygos:		Pakeistas katalogo pavadinimas
Alternatyvūs scenarijai		
1. Kataloge jau egzistuoja katalogas nauju vardu		1. Operacija atmetama 1.1. Parodomas informacinis pranešimas

2.9 lentelė. PA „Ištrinti katalogą“ aprašymas

PA „Ištrinti katalogą“		
Paketas: Failų sistema		
Tikslas: Pažymėti katalogą kaip ištrintą ir neberodyti failų sistemoje		
Aprašymas: Naudotojas gali pašalinti katalogą iš failų sistemos		
Prieš sąlyga:		Paleistas sistemos mazgas Sukurtas bent vienas katalogas
Sužadinimo sąlyga:		Pasirinktam katalogui parenkama ištrynimo funkcija
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Pasirenkamas katalogas 2. Pasirenkama katalogo trynimo funkcija		1. Atidaromos failo funkcijos 2. Sukuriama nauja failo ištrynimo transakcija
Po sąlygos:		Failas neberodomas failų sistemoje ir yra pašalintas iš failų saugyklos

2.10 lentelė. PA „Perkelti failą į katalogą“ aprašymas

PA „Perkelti failą į katalogą“		
Paketas: Failų sistema		
Tikslas: Perkelti failą iš vieno katalogo į kitą		
Aprašymas: Naudotojas gali perkelti pasirinktą failą į kitą katalogą		
Prieš sąlyga:		Paleistas sistemos mazgas Įkeltas bent vienas failas Sukurtas bent vienas katalogas
Sužadinimo sąlyga:		Pasirenkamas failas perkėlimui
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Pasirenkamas failas 2. Failas pažymimas perkėlimui 3. Atveriamas kitas katalogas 4. Pasirenkama failo perkėlimo funkcija		1. Atidaromos failo funkcijos 2. Įsimenamas failas perkėlimui 3. Atvaizduojamas katalogo turinys 4. Sukuriama nauja transakcija
Po sąlygos:		Failas perkeltas į naują katalogą
Alternatyvūs scenarijai		

1. Kataloge jau egzistuoja failas su perkeliamo failo pavadinimu	1. Operacija atmetama 1.1. Parodomas informacinis pranešimas

2.11 lentelė. PA „Peržiūrėti blokų grandinę“ aprašymas

PA „Peržiūrėti blokų grandinę“		
Paketas: Failų sistema		
Tikslas: Atvaizduoti visą grandinę peržiūrai		
Aprašymas: Kiekvienas naudotojas gali peržiūrėti visą blokų grandinę		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Pasirenkamas meniu punktas grandinės peržiūros langui atverti
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	Peržiūrėti bloką
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Naudotojas pasirenka meniu punktą blokų grandinės peržiūrai		1. Užkraunamas blokų grandinės sąrašas ir atvaizduojamas lentele
Po sąlygos:		Parodoma blokų grandinė

2.12 lentelė. PA „Peržiūrėti bloką“ aprašymas

PA „Peržiūrėti bloką“		
Paketas: Failų sistema		
Tikslas: Parodyti vienos bloko detales		
Aprašymas: Naudotojas gali peržiūrėti visą bloko informaciją		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Pasirinktas blokas peržiūrai
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	Peržiūrėti transakciją
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Naudotojas pasirenka bloką peržiūrai iš sąrašo		1. Atvaizduojama pasirinkto bloko informacija
Po sąlygos:		Parodoma pasirinkto bloko informacija

2.13 lentelė. PA „Peržiūrėti transakciją“ aprašymas

PA „Peržiūrėti transakciją“		
Paketas: Failų sistema		
Tikslas: Parodyti vienos transakcijos detales		
Aprašymas: Naudotojas gali vienos transakcijos, įrašytos į bloką, detales		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Pasirinkta transakcija peržiūrai
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	Peržiūrėti transakcijos turinį
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Naudotojas pasirenka transakciją iš bloko transakcijų sąrašo		1. Atvaizduojama pasirinktos transakcijos informacija
Po sąlygos:		Parodoma pasirinktos transakcijos informacija
Alternatyvūs scenarijai		
1. Naudotojas nesukūrė peržiūrimos transakcijos arba neturi teisės ją peržiūrėti		1. Nerodomas transakcijos turinys, rodoma tik bendra informacija

2.14 lentelė. PA „Peržiūrėti transakcijos turinį“ aprašymas

PA „Peržiūrėti transakcijos turinį“		
Paketas: Failų sistema		
Tikslas: Parodyti transakcijos turinį		
Aprašymas: Naudotojas gali peržiūrėti transakcijos turinį, jei turi teisę peržiūrėti		
Prieš sąlyga:		Paleistas sistemos mazgas Naudotojas turi teisę peržiūrėti transakcijos turinį
Sužadinimo sąlyga:		Pasirinkta transakcija peržiūrai
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Naudotojas pasirenka transakciją iš bloko transakcijų sąrašo		1. Atvaizduojama pasirinktos transakcijos informacija ir turinys
Po sąlygos:		Parodomas pasirinktos transakcijos turinys

2.15 lentelė. PA „Peržiūrėti nepatvirtintų transakcijų sąrašą“ aprašymas

PA „Peržiūrėti nepatvirtintų transakcijų sąrašą“		
Paketas: Failų sistema		
Tikslas: Parodyti nepatvirtintų transakcijų sąrašą		
Aprašymas: Naudotojas gali peržiūrėti nepatvirtintų transakcijų sąrašą		
Prieš sąlyga:		Paleistas sistemos mazgas Sistemoje sukurta bent viena nauja transakcija
Sužadinimo sąlyga:		Pasirenkamas meniu punktas nepatvirtintų transakcijų peržiūros langui atverti
Aktorius:		Naudotojas
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Naudotojas pasirenka meniu punktą nepatvirtintų transakcijų peržiūros langui atverti		1. Parodomos naudotojo sukurtos ir iš sistemos gautos nepatvirtintos transakcijos
Po sąlygos:		Parodomos nepatvirtintos transakcijos

2.16 lentelė. PA „Prisijungti prie failų sistemos“ aprašymas

PA „Prisijungti prie failų sistemos“		
Paketas: Blokų grandinės mazgas		
Tikslas: Prisijungti prie failų sistemos tinklo ir gauti blokų grandinę		
Aprašymas: Sistemos mazgas turi prisijungti prie kitų pasiekiamų tinklo mazgų ir gauti aktualią blokų grandinę		
Prieš sąlyga:		Sistemoje įdiegta failų sistemos programa
Sužadinimo sąlyga:		Naudotojas paleidžia failų sistemos programą
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	Generuoti raktų porą
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Paleidžiama programa 2. Klientas validuojamas tinkle 3. Gaunama blokų grandinė iš tinklo		1. Bandoma užmegzti ryšį su vienu iš pagrindinių mazgų
Po sąlygos:		Sėkmingai prisijungta prie tinklo ir gauta blokų grandinė

Alternatyvūs scenarijai	
1. Naujas klientas – dar nesukurta raktų pora	1. Sukuriama nauja raktų pora ir išsaugoma naudotojo kompiuteryje, bandoma jungtis iš naujo

2.17 lentelė. PA „Generuoti raktų porą“ aprašymas

PA „Generuoti raktų porą“		
Paketas: Blokų grandinės mazgas		
Tikslas: Sugeneruoti naują raktų porą naudotojui		
Aprašymas: Sistema turi sugeneruoti naują kriptografinių raktų porą klientui, jei jis jungiasi pirmą kartą		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Naudotojas paleidžia failų sistemos programą
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Paleidžiama sistema, bet nėra kliento raktų poros		1. Sukuriama nauja raktų pora
Po sąlygos:		Sėkmingai sukurta nauja raktų pora

2.18 lentelė. PA „Sukurti transakciją“ aprašymas

PA „Sukurti transakciją“		
Paketas: Blokų grandinės mazgas		
Tikslas: Sukurti naują transakciją		
Aprašymas: Sistema gali sukurti naują transakciją, kuri bus paruošta įrašyti į bloką		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Naudotojas inicijuoja operaciją sistemoje, kuri turi išsaugoti tam tikrus duomenis blokų grandinėje
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	Išsiųsti failą į saugyklą Išsiųsti transakciją
	Apimami PA	Užšifruoti duomenis Validuoti transakciją
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Inicijuojama transakcijos kūrimo operacija 2. Užšifruojami transakcijos duomenys 3. Suteikiamos transakcijos peržiūros teisės 4. Transakcija pasirašoma 5. Transakcija validuojama 6. Transakcija perduodama įrašymui ir išsiunčiama žinomiems mazgams		-
Po sąlygos:		Transakcija sukurta ir įrašyta į nepatvirtintų transakcijų baseiną
Alternatyvūs scenarijai		
1. Nėra žinomi reikalingi kriptografiniai raktai 2. Transakcija nevalidi		1 - 2. Operacija atmetama

2.19 lentelė. PA „Validuoti transakciją“ aprašymas

PA „Validuoti transakciją“		
Paketas: Blokų grandinės mazgas		
Tikslas: Patikrinti transakcijos duomenų tikslumą		
Aprašymas: Sistema tikrina transakcijos duomenis pagal įvairius kriterijus ir pagal tai ją priima arba atmeta		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Sistema sukuria transakciją arba gauna ją iš kito mazgo
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Sistema nuosekliai tikrina transakcijos informaciją		1. Transakcija atmetama arba priimama
Po sąlygos:		Transakcija priimta arba atmesta

2.20 lentelė. PA „Užšifruoti duomenis“ aprašymas

PA „Užšifruoti duomenis“		
Paketas: Blokų grandinės mazgas		
Tikslas: Užšifruoti grandinėje saugomus duomenis		
Aprašymas: Duomenys turi būti šifruojami tam, kad turinį galėtų peržiūrėti tik tie naudotojai, kuriems jis yra skirtas		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Sukuriama nauja transakcija
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	Išsiųsti failą į saugyklą
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Sukuriama nauja transakcija 2. Transakcijos turinys užšifruojamas		-
Po sąlygos:		Transakcijos duomenys užšifruoti

2.21 lentelė. PA „Išsiųsti failą į saugyklą“ aprašymas

PA „Išsiųsti failą į saugyklą“		
Paketas: Blokų grandinės mazgas		
Tikslas: Išsaugoti fizinius failo duomenis išorinėje saugykloje		
Aprašymas: Failai nėra saugomi blokų grandinėje tiesiogiai, taigi juos reikia išsaugoti išorinėje saugykloje		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Inicijuota failo įkėlimo operacija
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Inicijuojama failo įkėlimo operacija 2. Sukuriama transakcija 3. Failo duomenys siunčiami į išorinę saugyklą		-
Po sąlygos:		Failas išsiųstas ir sėkmingai įrašytas išorinėje saugykloje

2.22 lentelė. PA „Priimti transakciją“ aprašymas

PA „Priimti transakciją“		
Paketas: Blokų grandinės mazgas		
Tikslas: Priimti transakciją iš kito mazgo		
Aprašymas: Mazgas turi priimti transakcijas iš kitų mazgų tinklo veikimui palaikyti		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Gauta nauja transakcija
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	Validuoti transakciją
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Priimti nauja transakciją iš kito tinklo mazgo		1. Sistema validuoja transakciją
Po sąlygos:		Transakcija priimta ir išsaugota nepatvirtintų transakcijų baseine
Alternatyvūs scenarijai		
1. Transakcijos duomenys neteisingi		1. Operacija atmetama

2.23 lentelė. PA „Išsiųsti transakciją“ aprašymas

PA „Išsiųsti transakciją“		
Paketas: Blokų grandinės mazgas		
Tikslas: Išsiųsti sukurtą transakciją kitiems mazgams		
Aprašymas: Mazgas turi išsiųsti naują transakciją kitiems tinklo dalyviams		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Sukurta nauja transakcija ir ji yra validi
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Sukuriama nauja transakcija 2. Transakcija validuojama 3. Transakcija išsiunčiama žinomiems tinklo mazgams		-
Po sąlygos:		Transakcija išsiųsta tinklo mazgams

2.24 lentelė. PA „Iškasti bloką“ aprašymas

PA „Iškasti bloką“		
Paketas: Blokų grandinės mazgas		
Tikslas: Iškasti naują bloką		
Aprašymas: Sistema turi iškasti naują bloką, kurį sudaro tam tikras kiekis dar nepatvirtintų transakcijų		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Egzistuoja bent viena nepatvirtinta transakcija
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	Validuoti bloką
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Tikrinamas nepatvirtintų transakcijų baseinas 2. Bandoma iškasti naują bloką		1. Parenkamos nepatvirtintos transakcijos, jei tokių yra 2. Tikrinama, ar tenkinama sukonfigūruota atitikimo sąlyga

Po sąlygos:	Iškastas naujas blokas
Alternatyvūs scenarijai	
1. Iškastas blokas nevalidus	1. Operacija atmetama ir kartojama, jei yra nepatvirtintų transakcijų

2.25 lentelė. PA „Priimti bloką“ aprašymas

PA „Priimti bloką“		
Paketas: Blokų grandinės mazgas		
Tikslas: Priimti iškasta bloką iš kito mazgo		
Aprašymas: Naujas blokas gali būti iškastas kito mazgo, taigi jį reikia priimti ir prijungti prie lokalios blokų grandinės		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Gautas naujas blokas iš kito tinklo mazgo
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	Validuoti bloką
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Gaunamas naujas blokas		1. Atliekama bloko validacija
Po sąlygos:		Blokas priimtas

2.26 lentelė. PA „Validuoti bloką“ aprašymas

PA „Validuoti bloką“		
Paketas: Blokų grandinės mazgas		
Tikslas: Patikrinti bloko duomenų tikslumą		
Aprašymas: Sistema tikrina bloko duomenis pagal įvairius kriterijus ir pagal tai jį priima arba atmeta		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Sistema iškasa bloką arba gauna jį iš kito mazgo
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	Įrašyti bloką į grandinę
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Sistema nuosekliai tikrina bloko informaciją		1. Blokas atmetamas arba priimamas
Po sąlygos:		Blokas priimtas arba atmestas

2.27 lentelė. PA „Įrašyti bloką į grandinę“ aprašymas

PA „Įrašyti bloką į grandinę“		
Paketas: Blokų grandinės mazgas		
Tikslas: Papildyti blokų grandinę nauju bloku		
Aprašymas: Sistema turi įrašyti naują bloką į grandinę		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Sukurtas naujas blokas arba gautas naujas blokas iš kito tinklo mazgo
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Gaunamas naujas blokas 2. Blokas validuojamas 3. Blokas įrašomas į blokų grandinę		-
Po sąlygos:		Blokas įrašomas į blokų grandinę

Alternatyvūs scenarijai	
1. Blokas nevalidus	1. Operacija atmetama

2.28 lentelė. PA „Išsiųsti bloką“ aprašymas

PA „Išsiųsti bloką“		
Paketas: Blokų grandinės mazgas		
Tikslas: Išsiųsti sukurtą arba persiųsti gautą transakciją kitiems tinklo mazgams		
Aprašymas: Sistema turi pasirūpinti blokų persiuntimu visame mazgų tinkle		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Sukurtas arba gautas naujas blokas
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Iškasamas arba gaunamas naujas blokas 2. Blokas išsiunčiamas kitiems žinomiems tinklo mazgams		-
Po sąlygos:		Blokas išsiųstas žinomiems tinklo mazgams

2.29 lentelė. PA „Iššifruoti duomenis“ aprašymas

PA „Iššifruoti duomenis“		
Paketas: Blokų grandinės mazgas		
Tikslas: Išgauti prasmingą informaciją iš transakcijos		
Aprašymas: Sistema turi išgauti prasmingą informaciją iš transakcijos turinio pasinaudojant kriptografiniais raktais		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Inicijuojama operacija transakcijos peržiūrai ar failo parsiuntimui
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Inicijuojama operacija transakcijos peržiūrai ar failo parsiuntimui 2. Surandama reikiama transakcija 3. Transakcijos turinys iššifruojamas pasinaudojant kriptografiniais raktais		-
Po sąlygos:		Duomenys iššifruoti ir paruošti naudojimui

2.30 lentelė. PA „Gauti failą iš saugyklos“ aprašymas

PA „Gauti failą iš saugyklos“		
Paketas: Blokų grandinės mazgas		
Tikslas: Gauti išsaugoto failo duomenis iš failų saugyklos		
Aprašymas: Sistema turi parsiųsti failo duomenis ir paruošti juos naudojimui		
Prieš sąlyga:		Paleistas sistemos mazgas
Sužadinimo sąlyga:		Inicijuojama failo atsiuntimo operacija
Aktorius:		Sistema
Susiję PA	**Išplečiantys PA**	-
	Apimami PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija**
1. Inicijuojama failo atsiuntimo operacija		-

2. Pagal failo transakcijos turinį randamas atitinkamas failas saugykloje ir yra parsiunčiamas į naudotojo kompiuterį	
Po sąlygos:	Failas atsiųstas iš saugyklos

2.1.2. Nefunkciniai reikalavimai

Sistemai keliami nefunkciniai reikalavimai aprašomi lentelėmis žemiau (2.31 - 2.34 lentelės). Pateikiamas kiekvieno reikalavimo aprašymas ir atitikimo kriterijus.

2.31 lentelė. Nefunkcinis reikalavimas failų sąrašo įkėlimo greičiui

Numeris	1.	**Tipas**	Nefunkcinis
Aprašymas	Failų sąrašo įkėlimas trunka ne ilgiau nei 2 sekundės.		
Atitikimo kriterijus	Atidarant failų naršymo langą, katalogo turinys turi būti pateikiamas per 2 sekundes ir greičiau. Taip pat šis kriterijus taikomas atliekant ir kitas failų sistemos operacijas bei naršant bet kokio gylio katalogų struktūroje.		

2.32 lentelė. Nefunkcinis reikalavimas duomenų atnaujinimui realiu laiku

Numeris	2.	**Tipas**	Nefunkcinis
Aprašymas	Duomenų ir vaizdo atnaujinimas realiu laiku.		
Atitikimo kriterijus	Atverti langai ir juose vaizduojama informacija atsinaujina automatiškai vos tik gavus naujus duomenis iš sistemos tinklo.		

2.33 lentelė. Nefunkcinis reikalavimas mazgo būsenos atvaizdavimui

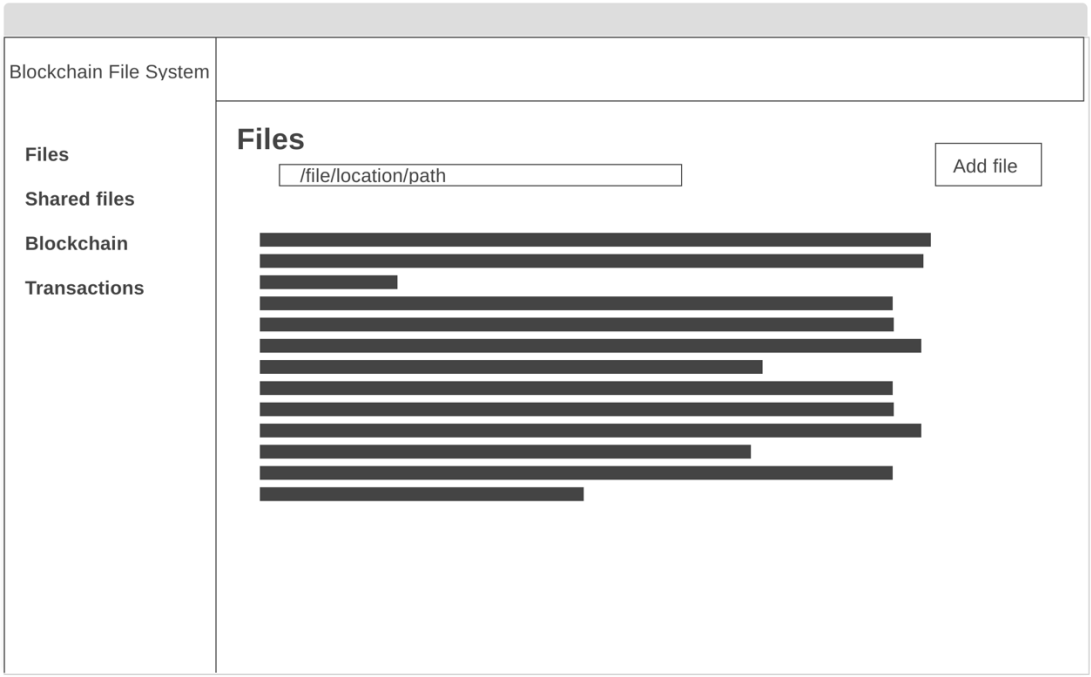
Numeris	3.	**Tipas**	Nefunkcinis
Aprašymas	Atvaizduojama kliento mazgo būsena.		
Atitikimo kriterijus	Grafinėje naudotojo sąsajoje matoma kliento mazgo būsena, kuri parodo, ar kliento mazgas tinkamai susinchronizuotas viso tinklo atžvilgiu.		

2.34 lentelė. Nefunkcinis reikalavimas realizavimo technologijoms

Numeris	4.	**Tipas**	Nefunkcinis
Aprašymas	Sistema realizuota technologijomis, kurias palaiko įvairios (populiariausios) operacinės sistemos.		
Atitikimo kriterijus	Sukurta sistema veikia Linux, Windows, macOS operacinėse sistemose.		

2.1.3. Rekomenduojama naudotojo sąsaja

Naudotojas sistema naudosis per tam tikrą naudotojo sąsają, taigi svarbu, kad būtų apibrėžti minimalūs reikalavimai, pagal kuriuos bus realizuojamas kuriamas produktas. Žemiau pateikiamas naudotojo sąsajos prototipo langas, kuris apibrėžia bendrą sistemos vaizdą ir pateikia gaires grafinės vartotojo sąsajos realizacijai. Nuspręsta taikyti standartinį išdėstymą: meniu kairėje, o atvertas langas užima visą likusį ekrano plotą. Pateikiamas vienas failų naršymo langas (2.4 pav.). Turinio dalį (vaizduojama juodomis linijomis) rekomenduojama realizuoti lentele, pateikiant detales apie failus ir katalogus.



2.4 pav. Rekomenduojama naudotojo sąsaja

2.2. Projekto sprendimo metodas

Šis skyrius skirtas aprašyti, kaip failų sistemoje veiks blokų grandinė, kokios failų sistemos operacijos turi būti realizuotos, ir kaip atrodys statinis bei dinaminis sistemos vaizdai.

2.2.1. Blokų grandinės taikymas

Failų sistema bus paremta blokų grandine. Ši technologija veiks kaip pagrindinė duomenų bazė failų sistemos duomenims (failų metaduomenims) saugoti. Priešingai nei bet kokia tradicinė centralizuota duomenų bazė, blokų grandinė veikia decentralizuotos sistemos principu, taigi jos duomenys yra saugomi daugelyje skirtingų nepriklausomų kompiuterių (sistemos tinklo mazgų). Duomenų teisingumą sprendžia kiekvienas mazgas sistemoje ir bendru nutarimu yra parenkama, kuri transakcija yra teisinga ir kuris blokas yra sekantis teisingas grandinės blokas.

Transakcija – tai duomenų vienetas sistemoje, kuris savyje saugo vienos operacijos duomenis. Transakcija kuriama ir siunčiama į tinklą naudotojui atlikus bet kokią operaciją, pvz.: įkėlus naują failą ar perkėlus jau esamą failą į naują katalogą. Turinį apibrėžiančios klasės ir su jomis susijusios operacijos aprašytos sekančiame skyriuje (2.2.2 Failų sistemos operacijos). Transakcijos turinys yra šifruojamas ir pasiekiamas tik turint prieigą prie šifravimo rakto. Kiekviena nauja transakcija turi būti patikrinama pagal numatytąsias taisykles, tokias kaip: ar transakcija dar neegzistuoja naujų transakcijų baseine, ar įrašytas siuntėjo kodas, ar teisingas parašas, ar teisinga maišos reikšmė (atliekamas papildomas perskaičiavimas ir sulyginama suskaičiuota reikšmė ir transakcijoje išsaugota reikšmė). Transakciją gali peržiūrėti tik tie asmenys, kuriems yra suteikiama teisė tai padaryti.

Transakcijos peržiūros teisės reguliuoja, kas gali perskaityti transakcijos turinį. Paprastai tai gali padaryti tik tas naudotojas, kuris inicijavo tam tikrą operaciją sistemoje ir sukūrė transakciją. Jeigu atliekamas veiksmas, kurio rezultatas aktualus ir kitam naudotojui, pavyzdžiui pasidalinama failu,

tuomet transakcijos turinį gali matyti ir kitas naudotojas. Transakcijos turinio iššifravimui reikalingas unikalus kriptografinis raktas. Kiekvienas naudotojas, kuris gali matyti transakcijos turinį, gauna atskirą kopiją, užšifruotą individualiu kliento viešuoju raktu. Tokiu būdu užtikrinama, kad transakciją pilnai peržiūrėti galės tik tie naudotojai, kurie turi reikiama raktą iššifravimui atlikti.

Naujai sukurtos transakcijos turi savo vietą ir yra laikomos nepatvirtintų transakcijų baseine. Tai sąrašas transakcijų, kurios dar nėra patekusios į jokį bloką ir yra paruoštos paimti į naujo bloko sukūrimą. Mazgas, sukūręs transakciją, išsaugo ją savo vietiniame transakcijų baseine ir pasidalina naująja transakcija visame tinkle, nes kiekvienas mazgas turi žinoti apie naujus įvykius sistemoje. Kiekvienas mazgas atliekas naujų transakcijų patikrinimą ir, jei nerandama klaidų, išsaugo transakciją savo vietiniame nepatvirtintų transakcijų baseine.

Blokų kasimas – tai sistemos veiksmas, kai iš naujų transakcijų yra formuojami nauji duomenų blokai ir jungiami į bendrą blokų grandinę. Tai esminė sistemos dedamoji dalis, nes ši operacija atsakinga už grandinės pildymą naujais blokais. Kiekvienas sistemos mazgas stengiasi kuo greičiau iškasti naują bloką ir išsiųsti jį į tinklą kitiems tinklo nariams. Šis veiksmas atliekamas stengiantis išspręsti matematinę užduotį pagal nustatytą sudėtingumą. Tai gali būti bloko maišos reikšmės, kuri turi prasidėti nustatytu kiekiu nulių, radimas. Sistema nuolat tikrina nepatvirtintų transakcijų baseiną ir, jei yra bent viena nauja transakcija, ji yra paimama ir pradedamas matematinės problemos sprendimas. Kiekvienos iteracijos metu didinama bloko *nonce* reikšmė, kuri yra viena iš dedamųjų dalių ieškant tinkamos bloko maišos reikšmės. Kai pasiekiamas tenkinamas rezultatas ir randamas sprendimas matematinei problemai, blokas laikomas iškastu ir gali būti jungiamas į blokų grandinę. Mazgas nauju bloku dalinasi visame tinkle. Kiekvienas mazgas, gavęs naują bloką, atlieka jo patikrinimą: turi sutapti paskutinio grandinės bloko maišos reikšmė ir naujame mazge nurodyta paskutinio grandinės bloko maišos reikšmė, transakcijų kiekis bloke negali viršyti maksimalaus numatyto transakcijų skaičiaus, turi būti žinomos visos bloke išsaugotos transakcijos, matematinės užduoties sudėtingumas lygis turi atitikti sistemoje sutartą lygį.

Blokų grandinė – tai sistemos duomenų bazė, į kurią duomenys įrašomi ir negali būti pakeisti per visą sistemos gyvavimo laiką. Nauji blokai jungiami vienas paskui kitą. Kiekvienas blokas turi nuorodą į prieš tai grandinėje einantį bloką. Tokios nuorodos neturi tik pirmasis grandinės blokas. Grandinės teisingumas užtikrinamas bendromis tinklo narių (mazgų) pastangomis. Mazgai kasa naujus blokus ir jais dalinasi su kitais tinklo dalyviais. Mazgo kasimas inicijuojamas automatiškai, kai egzistuoja bent viena nauja transakcija. Gali nutikti taip, kad du mazgai iškas sekanti grandinės bloką beveik tuo pačiu metu ir išsiųs jį į tinklą, tačiau pagal visas grandinės taisykles, teisingu gali būti pripažintas tik vienas. Įvykus tokio tipo konfliktams, grandinė gali tapti nestabili ir kai kurie mazgai gali tapti nepatikimais, taigi tolimesnis blokų tikrinimas ir grandinės plėtojimas jiems yra draudžiamas tol, kol nėra išsprendžiama kilusi problema. Šios kolizijos sprendžiamos bendromis jėgomis, kai nestabilią grandinės kopiją turintys mazgai prašo atsiųsti teisingą kopiją iš kitų mazgų. Gavus pakankamai informacijos, atliekamas grandinės taisymas nuo taško, kuriame kilo problema. Kai mazgas susitvarko vietinę grandinės kopiją ir gauna patvirtinimą iš tinklo, jis vėl gali dalyvauti sistemoje kaip patikimas narys, priimti naujus blokus, juos tikrinti ir jungti į blokų grandinę.

2.2.2. Failų sistemos operacijos

Failų sistemoje numatytos operacijos failų ir katalogų manipuliacijai atlikti: įkelti failą, kurti katalogą, pervardinti failą, perkelti failą į kitą katalogą, ištrinti failą, pervardinti katalogą, ištrinti failą, pervardinti katalogą, ištrinti katalogą ir pasidalinti failu. Kiekvienos operacijos metu sukuriama nauja transakcija, kuri yra patvirtinama visame tinkle ir padedama į transakcijų baseiną, iš kurio vėliau yra kasami nauji blokai ir jungiami į blokų grandinę. Kiekviena sukurta transakcija, priklausomai nuo atliktos operacijos failų sistemoje, turi skirtingą turinio struktūrą ir saugomą informacijos kiekį. Toliau, atskiruose skyreliuose, aprašoma kiekviena operacija ir vaizduojamos duomenų klasės, atsakingos už kiekvienos transakcijos saugomus duomenis.

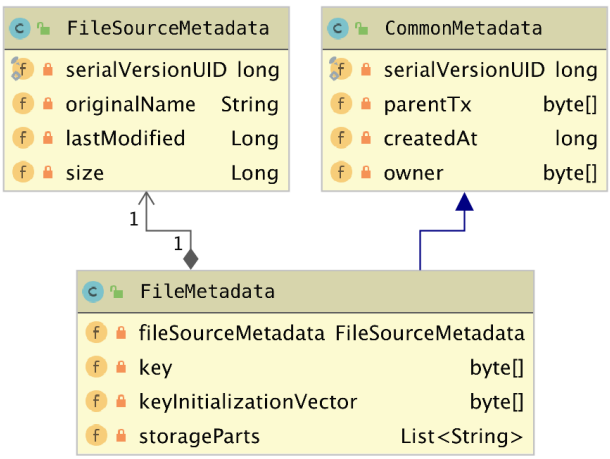
Tekste pateiktos klasių diagramos turi būti naudojamos realizuojant sistemą.

2.2.2.1. Įkelti failą

Failo įkėlimas į failų sistemą – tai svarbiausia operacija, kurios metu išsaugomas failas ir visa su juo susijusi informacija. Nuo šios operacijos prasideda failo gyvavimo ciklas failų sistemoje. Naudotojas iš savo įrenginio disko pasirenka failą ar kelis failus ir patvirtina atliekamą veiksmą. Tuomet sistema priima naują užklausą ir ją apdoroja. Prieš sukuriant transakcijas saugojimui blokų grandinėje ir išsiunčiant failus į saugyklą, turi būti atliekami tam tikri patikrinimai. Reikia tikrinti, ar įrašymo užklausa pateikta egzistuojančiam failų sistemos katalogui ir, ar toks failas dar nėra įkeltas šiame kataloge. Toliau kiekvienas failas turi būti apdorojamas individualiai – atliekamas failo suspaudimas, suspaustų duomenų padalinimas į daug lygių dalių, šifravimo rakto sukūrimas, kiekvienos dalies užšifravimas su sukurtu raktu, failo metaduomenų registravimas, užšifruotų failo dalių išsiuntimas į failų saugyklą, transakcijos, kurioje išsaugomi visi failo metaduomenys, sukūrimas, išsaugojimas naujų transakcijų baseine bei transakcijos išsiuntimas į tinklą kitiems mazgams tam, kad būtų atliekami papildomi patikrinimai ir iškasamas naujas blokas.

Failo suspaudimas reikalingas siekiant kuo optimaliau išnaudoti naudojamos saugyklos resursus ir taupyti tinklo srautą perduodant reikalingus duomenis. Taip pat tai apsunkina originalaus failo atgavimą neautorizuotoms šalims. Toliau turi būti atliekamas failo padalinimas išskaidant suspaustą failą į smulkesnes bylas – tai sustiprina originalaus failo apsaugą, nes tampa praktiškai neįmanoma atsekti failui priklausančių dalių ir jų eilės tvarkos, pagal kurią turėtų būti išgaunamas originalus failas.

Paskutinis ir pats svarbiausias failo apsaugos aspektas – šifravimas. Kiekviena failo dalis yra užšifruojama sugeneruotu simetriniu kriptografiniu raktu, o pats raktas užšifruojamas kliento viešuoju asimetriniu raktu. Užšifruotas raktas turi būti saugomas kartu su failo dalių nuorodomis į saugykloje saugomus failus. Ši informacija įrašoma prie failo metaduomenų ką tik sukurtoje naujoje transakcijoje. Duomenų struktūra (failo metaduomenys), kurie išsaugomi transakcijoje vaizduojami klasių diagrama (2.5 pav.). Klasė *FileSourceMetadata* skirta originalių failo parametrų saugojimui, tokių kaip: failo pavadinimas, paskutinio keitimo laiko žyma ir dydis. Klasė *CommonMetadata* atsakinga už bendrai sistemoje naudojamų duomenų saugojimą (pvz.: šią klasę gali paveldėti kitas duomenų objektas). Čia įrašoma informacija apie tėvinę transakciją (šiuo atveju tai failo katalogo, kuriame jis bus saugomas, katalogo transakcija), transakcijos sukūrimo laiko žyma bei savininko kliento kodas. Visa pastaroji informacija apjungiama pagrindinėje klasėje *FileMetadata*, kurioje saugomos failo dalių nuorodos į saugyklos failus, užšifruotas šifravimo raktas ir rakto inicijavimo vektorius.

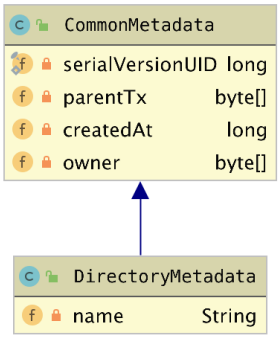


2.5 pav. Failo metaduomenų transakcijos struktūra

2.2.2.2. Kurti katalogą

Įkeltus failus patogu grupuoti atskiruose kataloguose, todėl katalogų kūrimo funkcija yra labai svarbi. Prie sistemos prisijungus naujam klientui, turi būti iš karto sukuriamas šakninis katalogas, kuriame toliau galima kurti kitus katalogus arba iš karto kelti norimus failus. Norint sukurti naują katalogą, naviguojama į kitą katalogą, kuriame kursime naująjį ir, inicijavus naujo katalogo kūrimo operaciją, įvedamas norimas vardas. Patvirtinus operaciją turi būti atliekamas patikrinimas, ar katalogas su tokiu pavadinimu dar neegzistuoja. Inicijavus šią operaciją, sistema sukuria naują

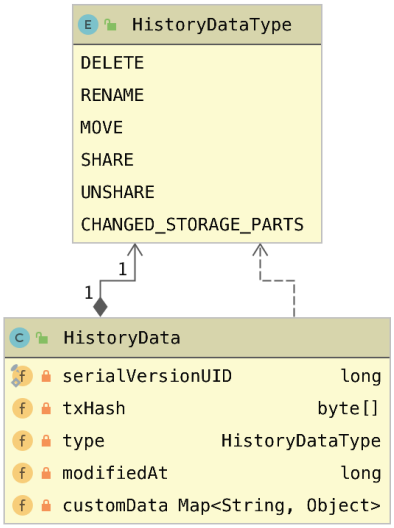
transakciją ir joje išsaugo naujo katalogo metaduomenis. Šių duomenų struktūrą vaizduoja klasių diagrama (2.6 pav.). *CommonMetada* klasė yra ta pati kaip ir kuriant failą, nes tai bendrieji duomenų laukai metaduomenims – čia išsaugoma tėvinio katalogo transakcijos nuoroda, savininkas bei transakcijos sukūrimo laiko žyma. Pagrindinė katalogo metaduomenų klasė *DirectoryMetadata* saugo katalogo pavadinimą.



2.6 pav. Katalogo metaduomenų transakcijos struktūra

2.2.2.3. Pervardinti failą

Sistemoje failų keisti tiesiogiai negalima, nes jie saugomi statiškai, tačiau galima keisti informaciją apie juos, pvz.: pakeisti failo pavadinimą. Kaip žinia, negalima pakeisti originalios failo transakcijos, kurioje yra pradiniai failo metaduomenys, taigi reikia kurti naują transakciją su atnaujinta informacija, susieti ją su pradine transakcija ir įrašyti į grandinę. Ši operacija atliekama pasitelkiant žemiau vaizduojamą klasių diagramą (2.7 pav.). Kiekvienas failo metaduomenų keitimo veiksmas atliekamas sukuriant transakciją su *HistoryData* objektu. Čia įrašoma nuoroda į susijusią transakciją, laiko žyma, duomenų tipas, kurį apibrėžia *HistoryDataType* išvardijimo klasė bei pasirinktinių duomenų objektas, kuriame saugoma specifinė informacija kiekvienam istorinio įrašo tipui. Failo pavadinimo keitimo atveju, į pasirenkamus duomenis įrašomas naujas failo pavadinimas, o tipas parenkamas *RENAME*. Jei failu dalinamasi su kitu naudotoju, šios transakcijos turinį galės peržiūrėti ir jis. Tokio tipo transakcijos yra labai lengvos ir neužima daug vietos grandinėje.



2.7 pav. Istorinio įrašo transakcijos struktūra

2.2.2.4. Perkelti failą į kitą katalogą

Failo perkėlimo operacija techniškai labai panaši į failo pervardinimą. Sukuriama transakcija *HistoryData* klasės pagrindu (2.7 pav.), parenkamas tipas *MOVE* ir į pasirenkamus duomenis įrašoma nuoroda į naujojo katalogo transakciją bei įrašoma nuoroda į susijusią transakciją kartu su sukūrimo laiko žyma.

2.2.2.5. Ištrinti failą

Tai mažai atminties resursų reikalaujanti operacija, kurios metu sukuriama transakcija klasės *HistoryData* pagrindu (2.7 pav.) ir parenkamas tipas *DELETE*. Daugiau jokie papildomi duomenys (išskyrus nuorodą į susijusią transakciją ir sukūrimo laiko žyma) neturi būti saugomi ir tokia transakcija jau paruošta įrašyti į blokų grandinę.

2.2.2.6. Pervardinti katalogą

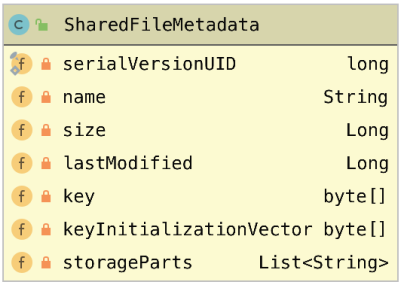
Ši operacija niekuo nesiskiria nuo failo pavadinimo keitimo. Sukuriama transakcija *HistoryData* pagrindu (2.7 pav.) ir įrašomas naujas katalogo pavadinimas. Parenkamas *RENAME* istorinių duomenų įrašo tipas bei įrašoma nuoroda į susijusią transakciją kartu su sukūrimo laiko žyma.

2.2.2.7. Ištrinti katalogą

Ši operacija tokia pati kaip ir failo trynimo atveju. Sukuriama transakcija klasės *HistoryData* pagrindu (2.7 pav.) ir parenkamas tipas *DELETE*. Daugiau jokie papildomi duomenys (išskyrus nuorodą į susijusią transakciją ir sukūrimo laiko žyma) neturi būti saugomi ir tokia transakcija jau paruošta įrašyti į blokų grandinę.

2.2.2.8. Pasidalinti failu

Pagal sistemos panaudos atvejus, kiekvienas naudotojas turi turėti galimybę dalintis savo įkeltais failais su kitais sistemos naudotojais. Šiam veiksmui atlikti reikia žinoti kito kliento identifikavimo kodą sistemoje. Ši operacija taip pat paremta *HistoryData* (2.7 pav.) objektu (su tipu *SHARE*), tačiau į papildomus duomenis įrašoma kur kas daugiau informacijos nei kitų operacijų metu. Kadangi tiesioginės prieigos prie originalios failo transakcijos suteikti negalima, kai kurie duomenys pakartojami kuriant dalinimosi transakciją: failo pavadinimas, dydis ir paskutinio keitimo data. Informacija saugoma *SharedFileMetadata* klasėje (2.8 pav.). Taip pat įrašoma nuorodos į failo dalis saugykloje ir raktas failui iššifruoti bei rakto inicijavimo vektorius. Raktas turi būti šifruojamas naudotojo viešuoju asimetriniu raktu. Transakcijoje įrašoma nuoroda į originalią failo transakciją bei kliento kodas, su kuriuo dalinamasi failu.

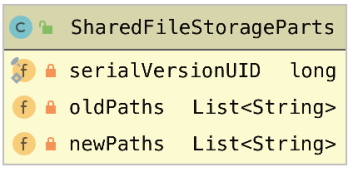


2.8 pav. Pasidalinamo failo metaduomenų transakcijos struktūra

2.2.2.9. Atšaukti failo dalinimąsi

Tai vienintelė operacija, kuri reikalauja dviejų transakcijų pilnam veikimo išpildymui. Taip yra todėl, kad viena transakcija atšaukia dalinimąsi sukuriant *HistoryData* objektą su tipu *UNSHARE* (2.7 pav.). Taip klientui indikuojama, kad failas nebepasiekiamas. Prie papildomų duomenų įrašomas kliento kodas, su kuriuo nebenorime dalintis failu.

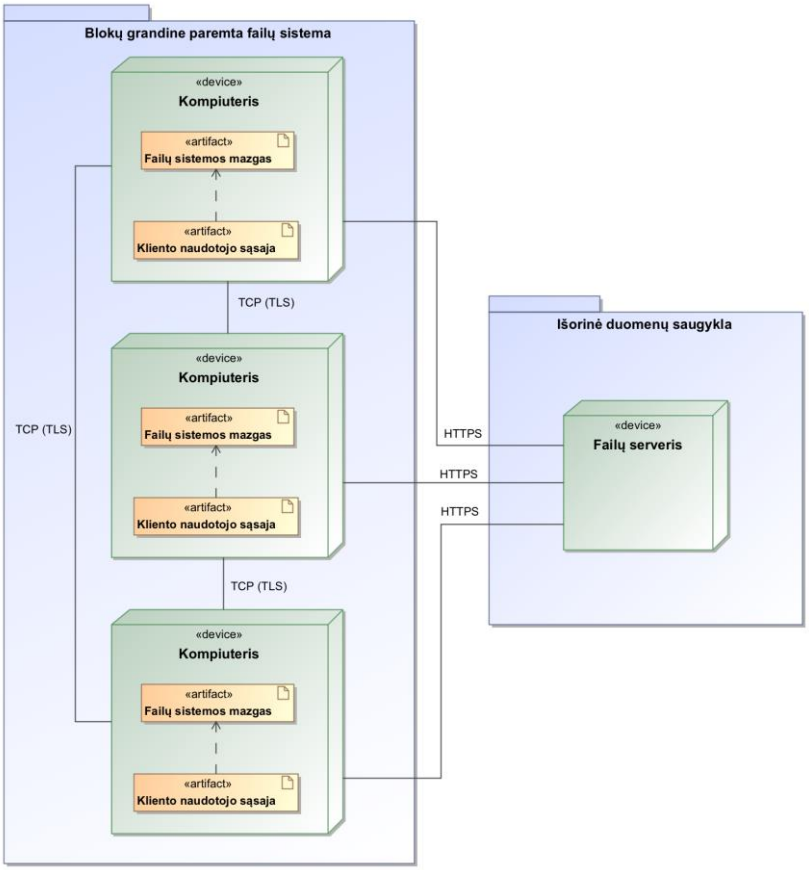
Kita transakcija yra skirta naujoms nuorodoms į failo saugyklą sukurti ir pasidalinti su klientais, kuriems vis dar yra suteikta prieiga prie failo. Kadangi blokų grandinėje įrašyta informacija gyvuoja amžinai ir negalima jos keisti, reikia imtis papildomų priemonių kai kurių veiksmų atlikimui. Šios operacijos vykdymo metu fiziškai keičiamos nuorodos į failo dalis saugykloje. Senosios ir naujosios nuorodos įrašomos į objektą *SharedFileStorageParts* (2.9 pav.), o šis įrašomas prie papildomų duomenų *HistoryData* objekte su tipu *CHANGED_STORAGE_PARTS*.



2.9 pav. Pakeistų failo nuorodų transakcijos papildomų duomenų struktūra

2.2.3. Statinis sistemos vaizdas

Statinį sistemos vaizdą vaizduoja išdėstymo (diegimo) diagrama (2.10 pav.). Vaizdas padalintas į du paketus – tai *blokų grandinės failų sistema* ir *išorinė duomenų saugykla*. *Blokų grandinės failų sistema* vaizduoja infrastruktūrą, kurioje gyvuoja blokų grandinė. Diagramoje matome tris įrenginius (kompiuterius), kurie palaiko tinklo veikimą, tačiau sistemoje bet kuriuo metu gali būti prisijungę *n* kompiuterių tarpusavyje bendraujančių TCP protokolo, apsaugoto TLS, pagalba. Viename kompiuteryje turi būti įdiegta failų sistemos programinė įranga, kurią sudaro failų sistemos mazgas, atsakingas už blokų grandinės veikimą, duomenų tikrinimą, apdorojimą ir saugojimą, ir kliento naudotojo sąsaja, kuria naudosis naudotojas norėdamas atlikti bet kokį veiksmą susijusį su failų sistema. Grafinė naudotojo sąsaja aptarnaujama tiesiai iš kompiuteryje veikiančio blokų grandinės mazgo per HTTPS protokolą interneto naršyklės pagalba. Kiekvienas sistemoje veikiantis mazgas turi prieigą prie išorinės duomenų saugyklos, kurioje saugomi naudotojų failai. Priklausomai nuo pačios failų saugyklos serverio, komunikacija gali vykti ne tik per HTTPS protokolą. Gali būti naudojami SCP, SFTP ar bet koks kitas failų perdavimui tinkamas protokolas. Pagrindinis reikalavimas – duomenų perdavimas turi būti saugus.

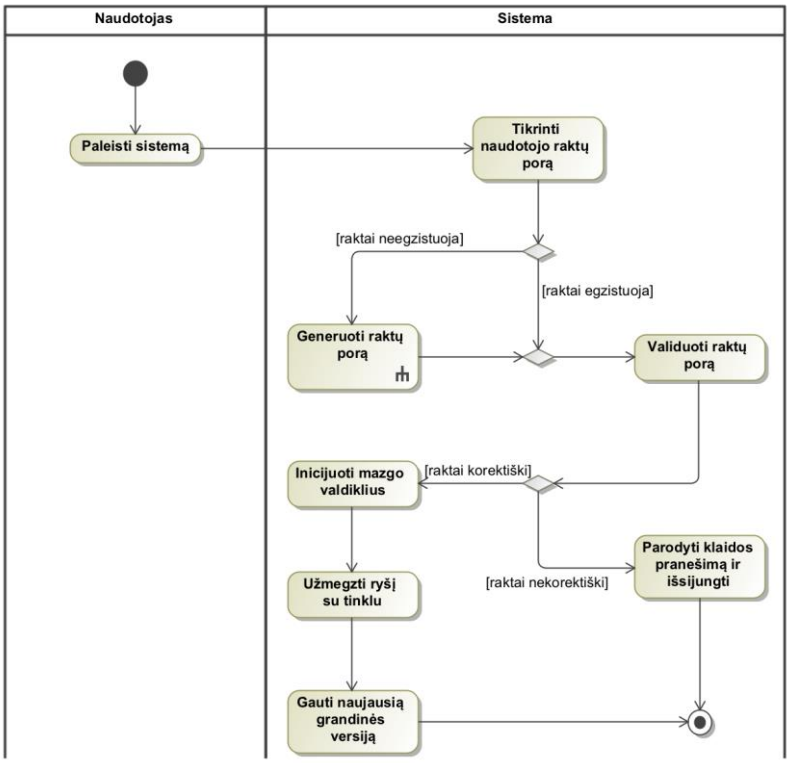


2.10 pav. Sistemos išdėstymo diagrama

2.2.4. Dinaminis sistemos vaizdas

Dinaminis sistemos vaizdas pateikiamas veiklos ir sekų diagramomis. Pateikiamos kelios esminės sistemos operacijos, detalizuojama jų atlikimo veiksmų seka.

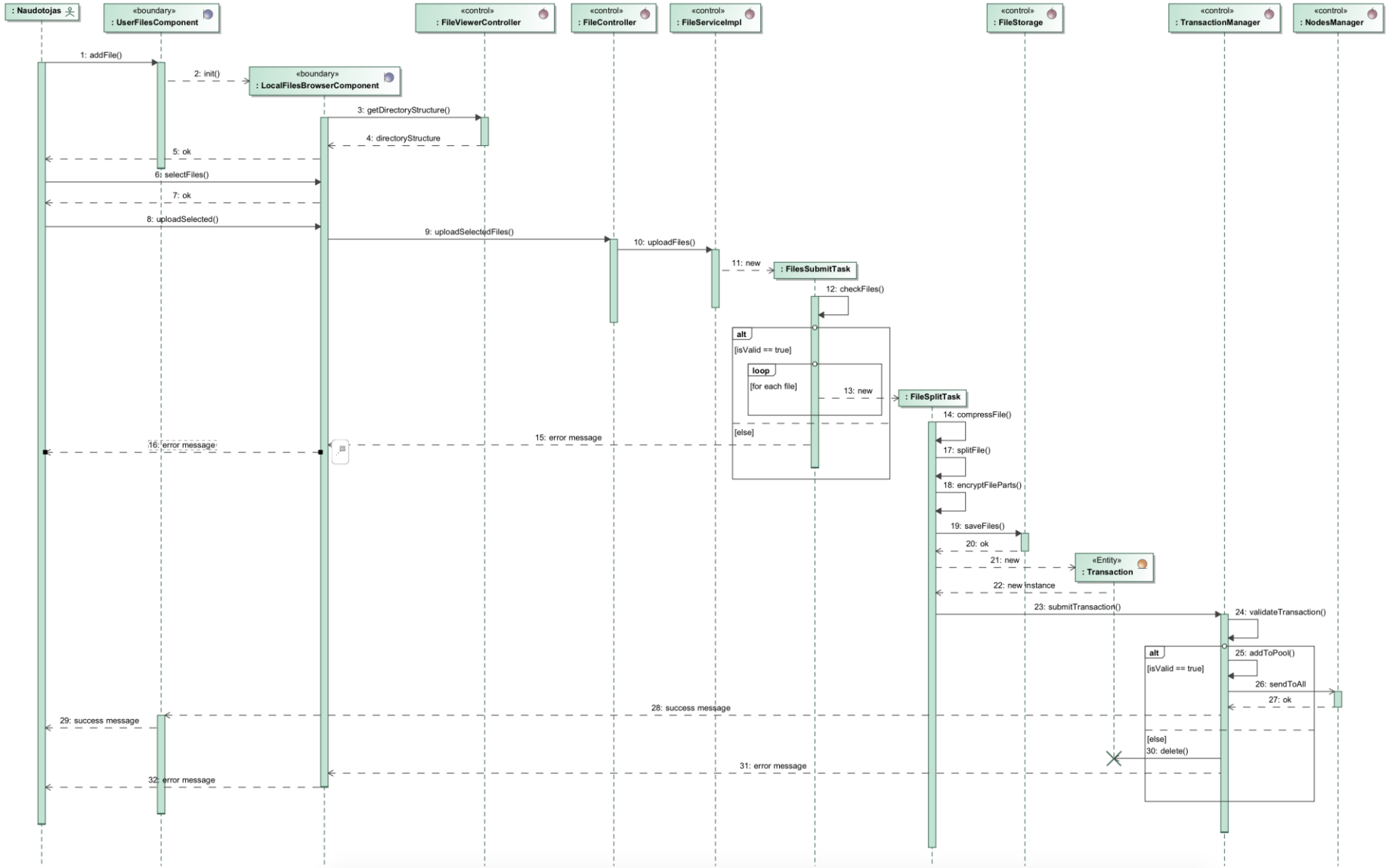
Prisijungimą prie failų sistemos inicijuoja naudotojas, kai paleidžia programą savo kompiuteryje. Prieš sistemai atliekant valdiklių ir kitų paslaugų paleidimą, atliekamas naudotojo raktų poros patikrinimas. Galimos dvi tolimesnės sekos: raktai egzistuoja arba ne. Jei raktai neegzistuoja, inicijuojamas jų generavimo procesas. Toliau bet kuriuo atveju atliekamas raktų poros validavimas sukuriant atsitiktinius duomenis, pasirašant juos privačiuoju raktu ir parašą tikrinant su viešuoju raktu. Jei šis veiksmas atliekamas sėkmingai – raktai laikomi tinkami naudoti. Nesėkmės atveju – naudotojas informuojamas klaidos pranešimu ir programa baigia darbą. Toliau vyksta sistemos mazgo valdiklių inicijavimas ir ryšio užmezgimas su kitas tinklo dalyviais. Visų pirma ryšys užmezgamas su numatytaisiais pagrindiniais mazgais. Jei jie neturi laisvos vietos susijungti su nauju mazgu, prašymą prisijungti siunčia kitiems žinomiems tinklo mazgams. Tai vyksta tol, kol randamas mazgas, turintis laisvos vietos naujam ryšio užmezgimui. Prisijungus prie tinklo, sistema paprašo atsiųsti blokų grandinės kopiją iš tinklo. Procesas baigiamas sėkmingai gavus ir užkrovus blokų grandinės kopiją. Procesas pavaizduotas veiklos diagrama (2.11 pav.).



2.11 pav. Veiklos diagrama: prisijungimas prie sistemos

Failų įkėlimas pavaizduotas sekų diagrama (2.12 pav.). Viskas prasideda nuo naudotojo, kuris naudotojo sąsajoje pasirenka naujo failo įkėlimo veiksmą. Sistema gauna darbinio katalogo struktūrą ir pateikia langą, kuriama galima pasirinkti failą bei naršyti kituose vietinio disko kataloguose. Pasirinkus vieną ar kelis failus, veiksmas tvirtinamas atskiru mygtuku. Užklausa keliauja į failų valdiklį, kuris perima apdorojimą ir pradeda kitą veiklos etapą – perduoda užklausą failų apdorojimo paslaugai. Ši paslauga sukuria naują failų įkėlimo užduotį ir perduoda ją užduočių valdikliui. Čia patikrinamas kiekvienas failas, kurį norima įkelti. Kiekvienam failui sukuriama nauja užduotis pagrindiniam paruošimui siųsti į saugyklą ir transakcijai sukurti. Ši užduotis failą suspaudžia, padalina į lygias dalis, kiekvieną dalį užšifruoja ir išsiunčia į failų saugyklą. Sėkmingai išsaugojus failą, surenkami reikalingi metaduomenys ir sukuriamas naujas transakcijos objektas. Toliau, transakcija perduodama transakcijų valdikliui, kuriame įvyksta transakcijos patikrinimas. Jeigu yra klaidų, operacija atmetama ir naudotojui parodomas klaidos pranešimas. Sėkmingu atveju transakcija pridedama į nepatvirtintų (neiškastų) transakcijų baseiną ir išsiunčiama į sistemos tinklą kitiems mazgams. Galiausiai naudotojas informuojamas apie sėkmingą operacijos atlikimą.

Veiksmų sekų bet kuriai sistemos operacijai įvykdyti yra gana panaši, taigi pasirinkta pavaizduoti vieną sudėtingiausią operaciją.



2.12 pav. Sekų diagrama: failo įkėlimas

2.2.5. Duomenų saugumo sprendimas

Kiekvienas sistemos tinko narys (mazgas) turi turėti unikalią kriptografinę asimetrinio šifravimo raktų porą – privatųjį ir viešąjį raktus. Šie raktai skirti naudotojo identifikavimui failų sistemoje, duomenų šifravimui ir pasirašymui atlikti. Paleidus sistemą, atliekamas raktų patikrinimas. Jei sistema paleidžiama pirmą kartą, raktai sukuriami automatiškai ir išsaugomi naudotojo kompiuteryje. Kitą kartą paleidžiant sistemą, raktai aptinkami automatiškai ir įkraunami naudojimui. Jeigu tas pats naudotojas nori pasiekti savo failus kitame kompiuteryje, privaloma turėti savo raktų porą ir nurodyti jų vietą kompiuteryje prieš paleidžiant sistemą. Raktų teisingumas gali būti tikrinamas atliekant atsitiktinę pasirašymo užduotį, kai privačiuoju raktu pasirašoma sugeneruoti duomenys ir viešuoju raktu parašas patvirtinamas.

Kiekviena sukurta transakcija turi būti pasirašoma naudotojo privačiuoju raktu. Parašui reikia surinkti tam tikrus duomenis iš transakcijos, kurie bus pasirašomi. Geriausia, kad kiekvienas duomenų laukas, esantis transakcijoje, būtų naudojamas pasirašymo procese. Tai yra: turinys, prieigos teisių duomenys, kliento, sukūrusio transakciją, kodas, transakcijos turinio raktas bei rakto inicijavimo vektorius. Privačiuoju raktu pasirašytus duomenis reikia išsaugoti transakcijoje. Kiekvienas tinklo mazgas, atlikdamas naujos transakcijos patikrinimą, atliekama parašo patikrinimą su kūrėjo viešuoju raktu.

Transakcijos turinys turi būti apsaugotas, nes ne kiekvienas tinklo narys turi turėti prieigą prie pagrindinių transakcijoje išsaugotų duomenų. Kadangi transakcijos turinys gali būti pakankamai didelis, asimetrinės kriptografijos raktu užšifruoti gali nepavykti. Šiam tikslui, kiekvienai transakcijai generuojamas atskiras simetrinės kriptografinis raktas. Šiuo raktu atliekamas transakcijos turinio šifravimas, o pats raktas užšifruojamas naudotojo viešuoju raktu ir išsaugomas transakcijoje. Tokiu būdu, norint peržiūrėti transakcijos turinį, reikia turėti kliento privatųjį raktą, iššifruoti specialų simetrinės kriptografijos raktą ir tik tada su gautu raktu iššifruoti transakcijos turinį.

Pagrindinis duomenų vienetas, kurį reikia apsaugoti failų sistemoje – tai naudotojo failas. Naudotojas, patalpinęs failą į failų sistemą, turi būti užtikrintas, kad tik jis vienas galės jį atgauti originalioje formoje. Šiam veiksmui atlikti turi būti naudojama seka tam tikrų operacijų, kurias atlikus vieną po kitos, bus užtikrinamas duomenų saugumas. Pateikus failą įkėlimui, visų pirma reikia atlikti jo suspaudimą. Tai ne vien prisidės prie saugumo sprendimo, tačiau bus taupomi ir failų sistemos resursai reikalaujant mažiau fizinės vietos failų saugykloje debesyje. Toliau, suspaustas failas padalinamas į vienodo dydžio dalis (atskirus failus). Tokiu būdu atsitiktinai failai neturės jokios prasmės bei siekiant atgauti originalų failą reikės žinoti dalių eiliškumą. Kiekviena dalis turi būti užšifruota. Šiam veiksmui atlikti, sugeneruojamas naujas simetrinės kriptografinis raktas ir juo užšifruojama kiekviena unikali failo dalis. Slaptasis raktas turi būti saugomas grandinėje prie failo metaduomenų. Tam, kad juo pasinaudoti galėtų tik tas naudotojas, kuris jį sukūrė, prieš išsaugant failo raktą į transakciją, atliekamas jo užšifravimas su kliento viešuoju raktu. Visos užšifruotos failo dalys saugiai perduodamos į failų saugyklą debesyje (priklausomai nuo pasirenkamos saugyklos implementacijos ir naudojamo protokolo) ir baigiama formuoti transakcija su visa reikalinga failo metaduomenų informacija: nuorodos į failo dalis saugykloje ir užšifruotas šifravimo raktas.

Tinklo mazgai nuolat dalinasi informacija tarpusavyje – perduoda transakcijas, blokus bei kitus sistemos veikimui reikalingus signalus. Nesvarbu, kokio svarbumo tai informacija, kiekvienas perduotas bitas turi saugiai pasiekti savo tikslą, išvengiant bet kokio tarpinio įsikišimo, duomenų pakeitimo ir praradimo. Numatytuoju atveju, bendravimas tinkle vyksta TCP/IP protokolu, svarbu užtikrinti, kad duomenys adresatą iš tiesų sėkmingai pasiekė. Tačiau tiesiogiai siųsti duomenis tokiu būdu nėra saugu, nes jie atvirai perduodami viešąja erdve, o tai nėra saugu, dėl galimo įsikišimo perdavimo metu. Komunikacijos saugumui užtikrinti turi būti naudojamas TLS (angl. *Transport Layer Security*) protokolas. Kiekvienas mazgas sistemoje gali veikti kaip serveris ir klientas. Tai priklauso nuo to, iš kurios pusės atliekamas susijungimas. TLS dėka užtikrinamas privatus ryšys, nes naudojama simetrinė kriptografija perduodamų duomenų šifravimui. Kiekvienam susijungimui generuojama nauja raktų pora, pagrįsta bendrąja paslaptimi, dėl kurios yra susitariama sesijos užmezgimo pradžioje. Mazgai sutaria, kokį kriptografinį algoritmą naudoti viso bendravimo metu. Toks susijungimas būtų

laikomas patikimu ir saugiu. Rekomenduojama naudoti TLS 1.3 dėl įvairių patobulinimų ir saugumo sustiprinimo lyginant su ankstesne TLS 1.2 versija.

2.3. Išvados

Pagal atliktos analizės išvadas buvo pasiūlytas metodas saugiam informacijos saugojimui naudojant blokų grandinių technologiją. Šis metodas išnaudoja blokų grandinę kuriant decentralizuotos failų sistemos branduolį. Blokų grandinė nėra skirta saugoti dideliems kiekiams informacijos, taigi įrašyti galima tik informaciją apie failų sistemą ir joje saugomus failus bei katalogus. Konkrečios duomenų bylos turi būti saugomos atskiroje išorinėje saugykloje, kurios paskirtis tik laikyti užšifruotus ir išskaidytus failus. Visų operacijų, atliktų veiksmų, failų metaduomenų istorija turi būti saugoma blokų grandinėje kartu su nuorodomis į konkrečius failus ir taisyklėmis, kaip juos gauti ir grąžinti į pirminę būseną.

Metodo savybės ir charakteristikos saugiam informacijos saugojimui:

- sistema decentralizuota, nėra vieno tiesos šaltinio, sprendimai priimami autonomiškai tarp visų tinkle veikiančių mazgų;
- paskirstyta sistema užtikrina pasiekiamumą;
- negalima keisti įrašytos informacijos, matoma visa įvykių istorija;
- duomenų patikrinimą atlieka visi sistemos tinklo nariai, taip užtikrinamas vientisumas;
- konfidencialumas užtikrinamas elektroniniu parašu ir pasitelkiant papildomas kriptografines priemones šifruojant transakcijų bei failų turinį.

Sistemai surinkti ir apibrėžti panaudos atvejai, išryškintas funkcionalumas. Aprašytos failų sistemos operacijos ir kaip jos turi būti realizuotos kuriamoje sistemoje. Pateikti statinis ir dinaminis sistemos vaizdai. Aprašytas duomenų apsaugos sprendimas naudojant duomenų suspaudimo ir kriptografines priemones.

3. BLOKŲ GRANDINE PAREMTOS FAILŲ SISTEMOS PROTOTIPO REALIZACIJA

Pagal parengtą failų sistemos projektą ir surinktus reikalavimus realizuotas sistemos prototipas – failų sistema paremta blokų grandinių technologija. Šiame skyriuje aprašomos realizacijos priemonės, panaudoti įrankiai, sukurti komponentai, klasės, problemų sprendimai, duomenų modelis, saugumo sprendimo realizacija, būsenos, paleidimo procesas ir naudotojo sąsajos vaizdas.

3.1. Realizacijos priemonės

Programinė įranga:

- IntelliJ IDEA 2020.1 Ultimate Edition – failų sistemos, blokų grandinės mazgo programavimas;
- Visual Studio Code 1.43.2 – kliento sąsajos programavimas;
- MagicDraw 19.0 – projektavimas ir modeliavimas (UML).

Programavimo kalbos:

- Java 11.0.5 – failų sistema, blokų grandinės mazgas (pagrindinė sistemos logika);
- TypeScript – kliento sąsajai realizuoti naudojant *Angular* karkasą.

Karkasai ir bibliotekos:

- Spring Boot 2.1.4 – Java karkasas;
- H2 atmintyje laikoma duomenų bazė;
- Angular 7.2.2 – JavaScript karkasas kliento grafinei sąsajai kurti.

Projekto valdymo priemonės:

- Maven 3 – projektui reikalingų trečiųjų šalių bibliotekų valdymas, programos ir jos paketų surinkimas į galutinį produktą;
- Git – projekto programinio išeities kodo versijos kontrolės palaikymui.

Operacinės sistemos:

- Windows 10 (kūrimas, testavimas, bandymai);
- macOS Catalina 10.15.3 (kūrimas, testavimas, bandymai).

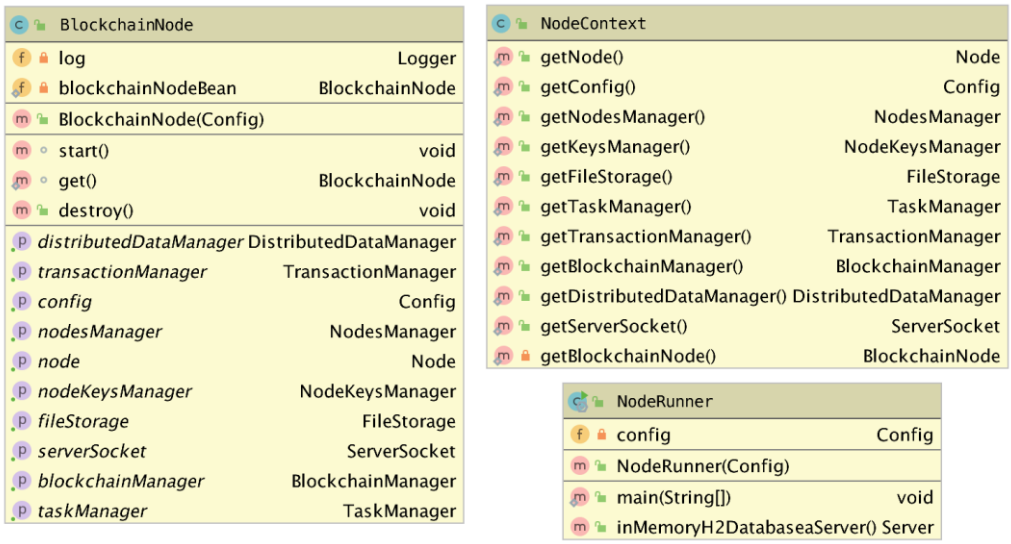
Papildomos priemonės:

- paruoštas grafinės sąsajos šablonas ir komponentai iš *CreativeTim* svetainės (https://www.creative-tim.com/).

3.2. Architektūra ir komponentai

Failų sistemos prototipas realizuotas 154 Java kodo klasėmis, kurias sudaro 7200 kodo eilučių. Toliau aprašomos pagrindinės klasės, paketai ir jų paskirtis.

Aukščiausio lygio pakete sukurtos pagrindinės failų sistemos mazgo klasės, nuo kurių pradedamas sistemos inicijavimas. Klasė *BlockchainNode* (3.1 pav.) atsakinga už visų valdiklių ir reikalingų paslaugų paleidimą ir laikymą. Šioje klasėje sukuriama mazgo konfigūracija užkraunant įvairius nustatymus iš nuostatų failo, atveriama tinklo jungtis, paleidžiami užduočių, paskirstytų duomenų, raktų, mazgų, failų saugyklos, transakcijų ir blokų grandinės valdikliai. Sistemos darbas pradedamas kviečiant metodą *start()*, kuriame ir įvyksta išvardintų komponentų inicijavimas. Tai atliekama *NodeRunner* klasėje (3.1 pav.) iškart po to, kai *Java Spring* sukuria programos kontekstą ir galime gauti *BlockchainNode* komponentą sistemos paleidimui atlikti. Klasė *NodeContext* (3.1 pav.) yra atsakinga už valdiklių ir mazgo informacijos pasiekimą bet kurioje programinio kodo dalyje.



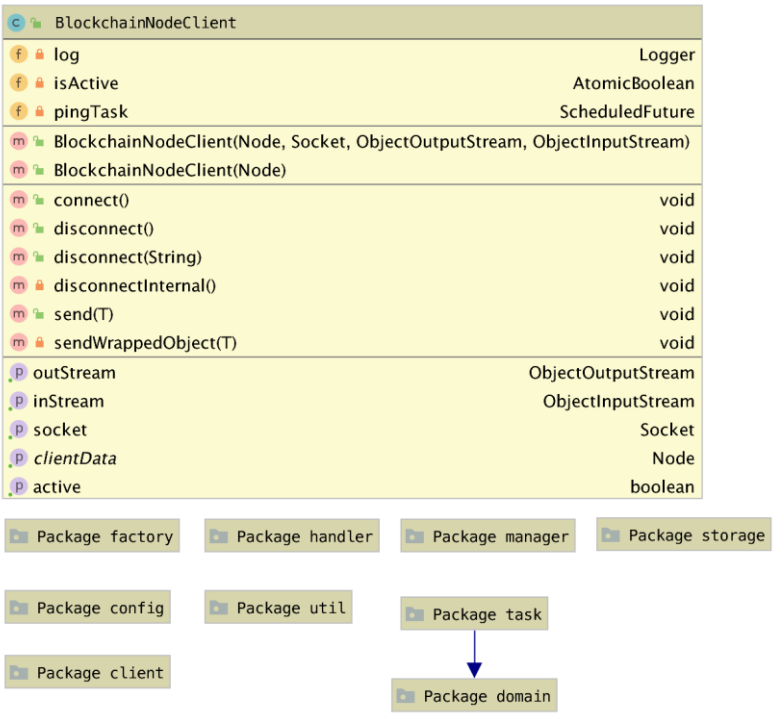
3.1 pav. Pagrindinės mazgo klasės

Sistemos konfigūracija užkraunama iš dedikuoto *application.properties* failo, kuriame keisti įvairius sisteminius parametrus. Konfigūraciją sudaro šie nustatymai:

- mazgo prievadas;
- kliento programos prievadas;
- ar įgalinti blokų kasimo mechanizmą;
- privačiojo ir viešojo raktų porų failų pavadinimai;
- failų saugyklos tipo parinkimas;
- programos gijų skaičius užduočių vykdymui;
- programos gijų skaičius suplanuotų ar nuolat vykstančių procesų vykdymui;
- įvairios parinktys išvedamiems pranešimams – programos vykdymo įrašų žurnale galima nerodyti tam tikrų pranešimų (pvz.: *ping* signalo išsiuntimo kitam mazgui, prisijungusių mazgų informacijos išvedimo, duomenų verifikavimo detalūs pranešimai ir t.t.);
- kas kiek laiko vykdyti suplanuotas ir nuolat vykstančias užduotis (*ping* signalo siuntimas prisijungusiems mazgas ryšio palaikymui, naujų mazgų aptikimas, neaktyvių mazgų pašalinimas iš atminties);
- atmintyje veikiančios duomenų bazės konfigūravimas: prievadas, prisijungimo adresas, naudotojas, slaptažodis;
- pagrindinių tinklo mazgų sąrašas, prie kurių bus bandoma prisijungti paleidus sistemą.

Toliau pagrindiniame pakete galima rasti klasę, kuri aptarnauja kitų prisijungusių tinklo mazgų poreikius – tai *BlockchainNodeClient* (3.2 pav.). Čia laikoma informacija apie kliento informaciją, tinklo sąsają ir aktyvumo indikaciją. Struktūra programoje tęsiama šiais paketais (3.2 pav.):

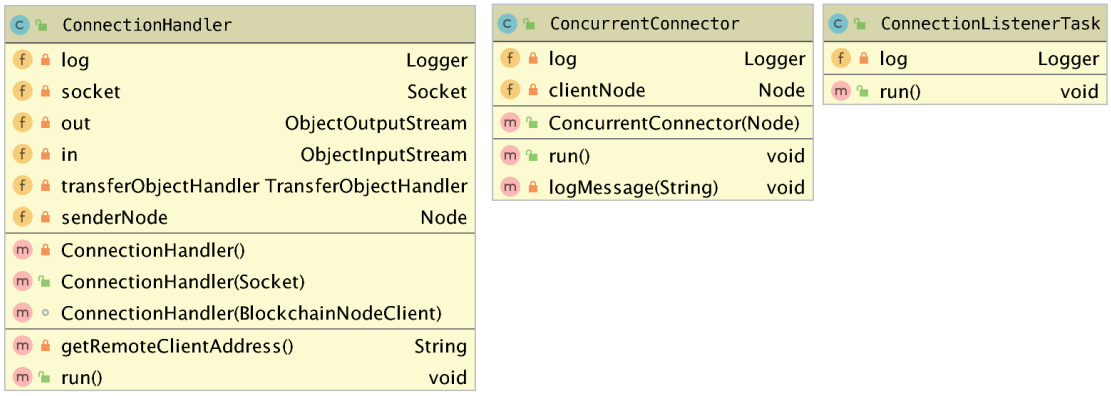
- *factory* – skirta klasėms, kurios kuria objektus (pvz.: transakcijas);
- *handler* – duomenų apdorojimo klasės;
- *manager* – valdiklių klasės;
- *storage* – saugyklų sąsajas implementuojančios klasės;
- *config* – konfigūracijos klasės;
- *util* – įvairios pagalbinės klasės (pvz.: atliekant šifravimą ar duomenų pasirašymą);
- *domain* – sistemos srities duomenų klasės;
- *task* – užduotis ir procesus implementuojančios klasės;
- *client* – kliento sąsaja aprūpinančios klasės.



3.2 pav. Mazgo kliento klasė ir sistemos paketai

Užduočių valdiklis (klasė *TaskManager*) (3.4 pav.) atsakingas už visų mazgo procesų paleidimą ir valdymą daugiagijoje aplinkoje. Sistemoje realizuoti dviejų tipų procesai: vykdomi vieną kartą nuo inicijavimo ir suplanuoti - pasikartojantys (periodiniai). Suplanuoti procesai – tai procesai, kurie nuolat vyksta fone nustatytu dažnumu. Šiuos procesus sudaro: naujų tinklo mazgų paieška, mazgo gyvybingumo palaikymas, neaktyvių mazgų išvalymas, blokų kasimas. Procesai, kurie yra vykdomi vieną kartą nuo jų sukūrimo: ryšio užmezgimas su kitu tinklo mazgu, perduodamų duomenų apdorojimas, naujų prisijungimų klausymasis, failų sistemos operacijų vykdymas (realizuotas atskiras procesas kiekvienai operacijai). Nesėkmingai pasibaigęs procesas, nutikus išimtiniam atvejui, gali būti kartojamas kelis kartus. Apie tai pranešama naudotojui kliento programoje.

Procesai, kurie atlieka tinklo operacijas, implementuoti klasėmis *ConnectionHandler* (priima duomenų objektus iš kitų tinklo narių), *ConcurrentConnector* (atlieka prisijungimą prie kito tinklo mazgo) ir *ConnectionListenerTask* (klausosi bandymų prisijungti ir perduoda valdymą procesui *ConnectionHandler*, kai gaunamas užklausimas naujam susijungimui) (3.4 pav.).



3.3 pav. Ryšio užmezgimo ir palaikymo klasės

Mazgų valdiklis (klasė *NodesManager*) (3.4 pav.) atlieka tinklo mazgų paiešką ir informacijos saugojimą apie mazgus, prie kurių prisijungiama. Ši klasė turi metodus mazgų pridėjimui, pašalinimui ir informacijos persiuntimui.

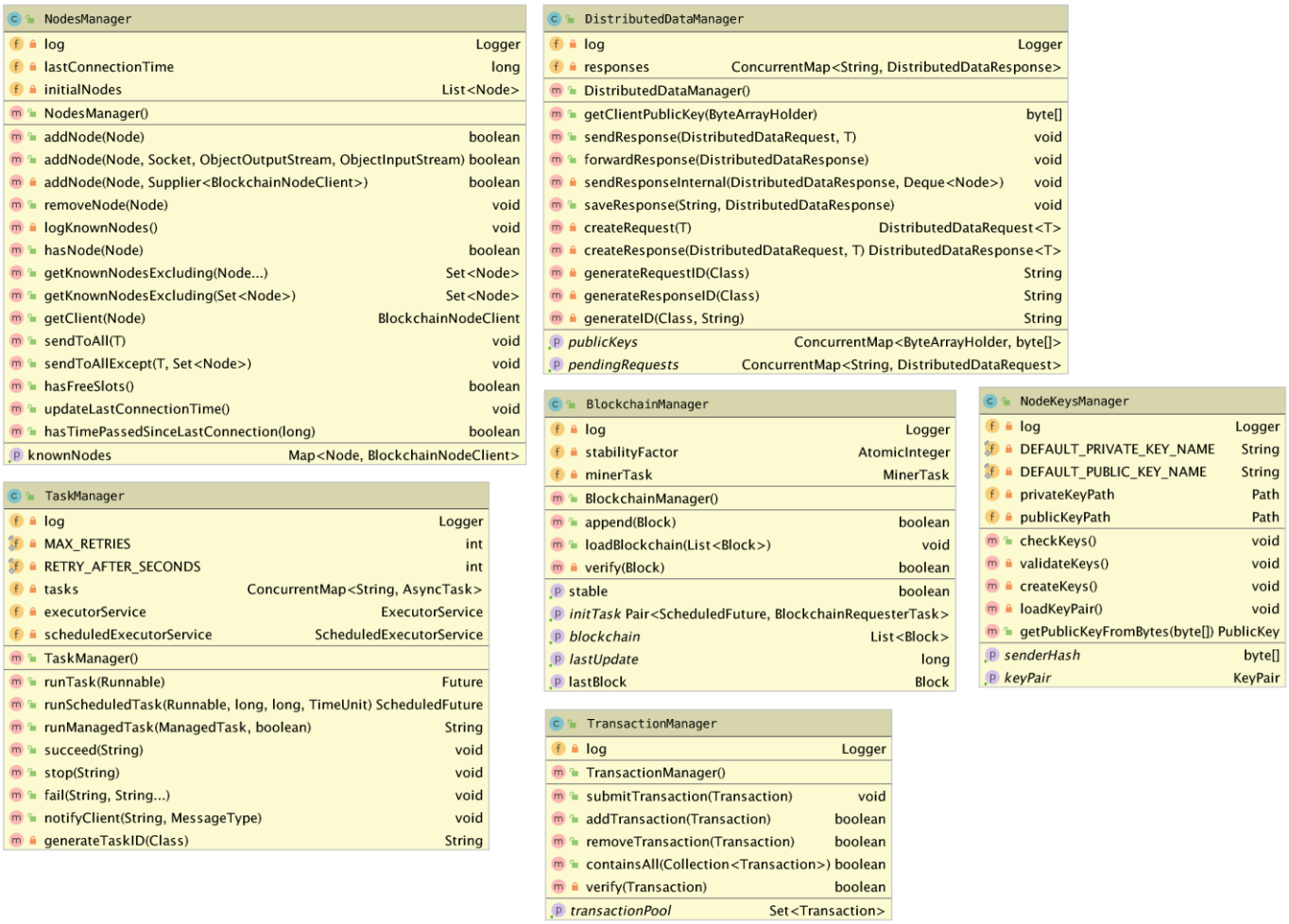
Paskirstytų duomenų valdiklis (klasė *DistributedDataManager*) skirtas duomenims gauti iš sistemos tinklo, kai reikiamus duomenis turi nebūtinai tiesioginį ryšį palaikantis mazgas. Šis valdiklis

sukuria užklausą duomenims gauti ir siunčia visiems žinomiems mazgams. Jei mazgas, gavęs užklausą, negali pats jos apdoroti (neturi prašomų duomenų), užklausą persiunčia tinklu toliau. Galiausiai reikiamas mazgas gauna užklausą ir grąžina atsakymą tuo pačiu keliu, kokiu užklausa atkeliavo. Šis valdiklis naudojamas kliento viešajam raktui gauti. Jeigu norime dalintis failu, turime gauti kito kliento viešąjį raktą duomenų šifravimui. Greičiausiai nėra užmegztas tiesioginis ryšys, taigi užklausa keliauja per tinklą ieškodama savo adresato.

Raktų valdiklis (klasė *NodeKeysManager*) (3.4 pav.) skirtas kliento raktų valdymui. Čia atliekamas raktų poros sukūrimas, patikrinimas ir užkrovimas. Valdiklis didžiąją dalį savo darbo atlieka sistemos paleidimo metu. Failų sistema gali naudotis tik patikrintą raktų porą turintis naudotojas. Jei jungiamasi pirmą kartą, šis valdiklis sukuria naują raktų porą ir išsaugo naudotojo kompiuterio diske. Kitą kartą jungiantis jau naudojami prieš tai sukurti raktai ir, po jų užkrovimo iš disko, atliekamas teisingumo patikrinimas atliekant pasirašymo testo užduotį. Tai užduotis, kai atsitiktiniai duomenys pasirašomi su kliento privačiu raktu ir bandoma juos verifikuoti naudojant viešąjį raktą. Jei testas sėkmingas, klientas tęsia darbą ir yra prijungiamas prie failų sistemos.

Vienas pagrindinių ir esminių valdiklių – transakcijų valdiklis (klasė *TransactionManager*) (3.4 pav.). Šis komponentas atsako už naujų transakcijų pridėjimą į nepatvirtintų transakcijų baseiną ir transakcijų patikrinimą. Kiekviena nauja transakcija turi praeiti visus patikrinimus, nes kitu atveju įvyks atmetimas. Patikrinimo metu žiūrima, ar tokia transakcija dar nebuvo gauta ir pridėta, ar turi būtiniausius duomenis (pvz.: siuntėjo kodą). Pagal siuntėjo kodą formuojama užklausa viešajam raktui gauti. Gavus raktą, patikrinamas transakcijos parašas.

Blokų grandinės valdiklis (klasė *BlockchainManager*) (3.4 pav.) atlieka blokų patikrinimą ir pridėjimą į blokų grandinę. Paleidus sistemą, šis valdiklis siunčia užklausą žinomiems mazgams ir prašo atsiųsti savo blokų grandinės kopiją. Gavus blokų grandin ę, atliekamas patikrinimas ir pradinis užkrovimas į atmintį. Kiekvienas blokas tikrinamas pagal numatytas taisykles: turi sutapti bloko nuoroda į paskutinį bloką su tikruoju paskutiniu bloku, turi būti teisingai apskaičiuota *Merkle* šaknis, transakcijų kiekis negali viršyti numatytojo dydžio vienam blokui, apie visas pridėtas transakcijas turi būti pranešta iš anksto ir jos turi būti pridėtos į nepatvirtintų transakcijų baseiną bei kasimo sudėtingumas negali būti žemesnis nei numatytasis. Įvykus nesutapimams ir konfliktams, skaičiuojamas stabilumo faktorius. Jei konfliktai toliau kartojasi, blokų grandinė laikoma nestabilia ir prašoma žinomų mazgų pagalbos sprendžiant kilusią problemą. Tinklo nariai atsiunčia savo blokų grandinės kopijas, pagal kurias atliekamas grandinės perrinkimas ir sinchronizacija.



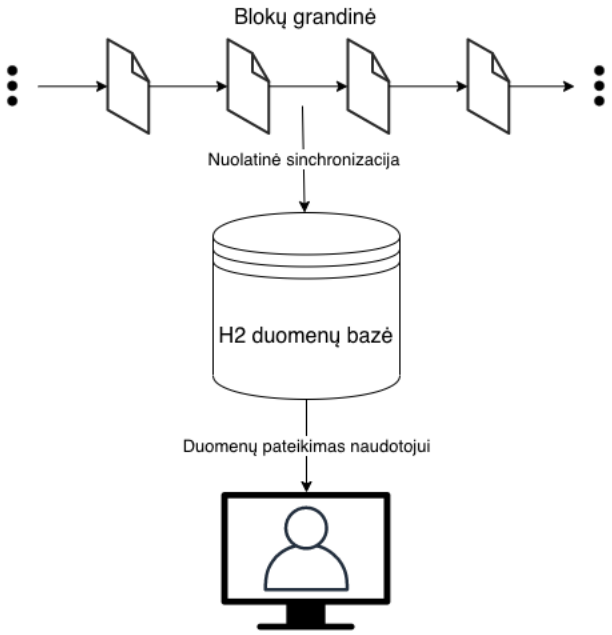
3.4 pav. Sistemos valdiklių klasės

3.2.1. Kliento spartinimas su atmintyje saugoma duomenų baze

Blokų grandinė nėra efektyvi, kai reikia daug kartų ir greitai peržiūrėti duomenis ar juos patikrinti. Šie veiksmai yra būtini, jei norima greitai pateikti informaciją naudotojui apie jo įkeltus failus ir sukurtus katalogus. Jei duomenų gavimui būtų naudojama blokų grandinė tiesiogiai, laikui bėgant sistema labai sulėtėtų ir galiausiai taptų sunkiai naudojama, nes blokų grandinė didėtų su laiku naudojantis failų sistema. Šiai problemai spręsti buvo panaudota atmintyje saugoma reliacinė duomenų bazė. Ji savyje laiko identišką blokų grandinės kopiją. Tokia duomenų bazė puikiai tinka greitai gauti reikiamus duomenis ir yra išvengiamas poreikis iteruoti per visus blokus grandinėje ir juose išsaugotas transakcijas. Kiekviena katalogo ar konkretaus failo peržiūra reikalauja surinkti visą informaciją iš naujo ir nuolatinis iteravimas grandine sukeltų smarkų spartos mažėjimą.

Viena populiariausių tokio tipo duomenų bazių – „H2“. Lentelės ir reikalinga struktūra sukuriama kiekvieną kartą startavus failų sistemą. Kai sistema gauna blokų grandinės kopiją iš tinklo, tuoj pat yra padaroma grandinės kopija ir į dedikuotą atmintyje saugomą duomenų bazę. Toliau su kiekvienu veiksmu, nauja transakcija ar bloku, šios duomenų bazės būsena yra sinchronizuojama pagal tikrąją blokų grandinę, kurios duomenų teisingumas yra užtikrinamas visame sistemos tinkle tarp veikiančių mazgų.

Duomenų bazė veikia kaip tarpininkas failų sistemos spartinimui. Šis mechanizmas vaizduojamas 3.5 paveiksle.



3.5 pav. Kliento spartinimo mechanizmas

3.3. Duomenų modelis

Duomenų saugojimas paremtas keliomis pagrindinėmis duomenų klasėmis: *Block*, *Transaction*, *TransactionAccess*. Šių klasių objektai – tai pagrindiniai komponentai formuojant blokų grandinę.

Transakcija skirta bet kokios failų sistemos operacijos, kurią atlieka naudotojas, duomenims saugoti. Transakciją sudaro duomenų laukai aprašyti 3.1 lentelėje.

3.1 lentelė. Transakcijos struktūra

Duomenų laukas	Aprašymas
hash	Transakcijos maišos reikšmė
senderHash	Siuntėjo kodas
signature	Transakcijos parašas
payload	Turinys
timestamp	Laiko žyma
canView	Sąrašas naudotojų, kurie gali peržiūrėti transakcijos turinį

Transakcijos objektas (3.6 pav.) taip pat turi ir keletą metodų kai kuriems duomenims apskaičiuoti. Šių metodų dėka gaunama transakcijos maišos reikšmė ir surenkami duomenys, kurie naudojami pasirašant transakciją. Toliau pateikiamas šių metodų išeities kodas – pirmasis metodas (*calculateHash*) apskaičiuoja maišos reikšmę, o antrasis (*getSignableData*) surenka duomenis parašui.

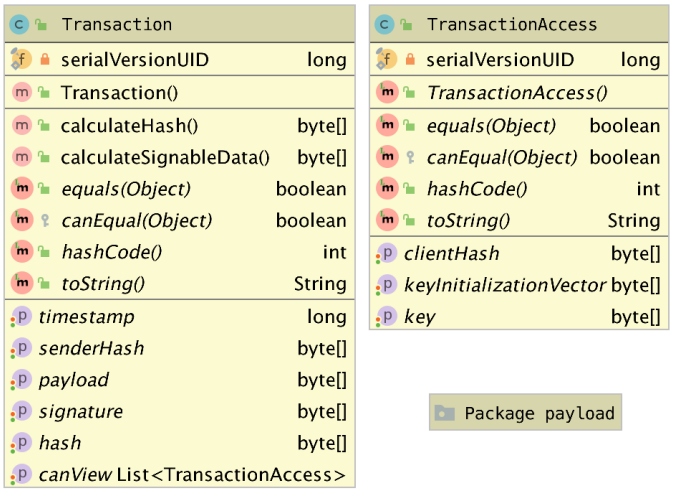
```
public byte[] calculateHash() {
   byte[] hashableData = ArrayUtils.addAll(payload, senderHash);
   hashableData = ArrayUtils.addAll(hashableData, signature);
   hashableData = ArrayUtils.addAll(hashableData, Longs.toByteArray(timestamp));

   for (TransactionAccess access : canView) {
      hashableData = ArrayUtils.addAll(hashableData, access.getClientHash());
      hashableData = ArrayUtils.addAll(hashableData, access.getKey());
      hashableData = ArrayUtils.addAll(hashableData, access.getKeyInitializationVector());
   }

   return DigestUtils.sha256(hashableData);
}
```

```
public byte[] getSignableData() {
    byte[] signableData = ArrayUtils.addAll(payload);

    for (TransactionAccess access : canView) {
        if (!Arrays.equals(access.getClientHash(), senderHash)) {
            continue;
        }
        signableData = ArrayUtils.addAll(signableData, access.getClientHash());
        signableData = ArrayUtils.addAll(signableData, access.getKey());
        signableData = ArrayUtils.addAll(signableData, access.getKeyInitializationVector());
        break;
    }

    return signableData;
}
```

Naudotojų, kurie gali peržiūrėti transakciją, duomenys saugomi *TransactionAccess* klasės objektuose (3.6 pav.). Jų struktūra skirta saugoti kliento kodui ir raktui transakcijos turinio iššifravimui (3.2 lentelė).

3.2 lentelė. Prieigos teisių duomenų struktūra

Duomenų laukas	Aprašymas
clientHash	Kliento kodas
key	Raktas transakcijos turiniui
keyInitializationVector	Rakto inicijavimo vektorius



3.6 pav. Transakcijos ir prieigos teisų klasės, ir transakcijos turinio klasių paketas

Klasės pakete *payload* sukurtos pagal klases sistemos projekto skyriuje, kuriame buvo aprašomos failų sistemos operacijos (2.2.2 Failų sistemos operacijos). Šios klasės atsakingos už transakcijų turinio struktūrą kuriant failą, katalogą ir atliekant kitas failų sistemai būdingas operacijas, kurių metu reikia išsaugoti informaciją blokų grandinėje sukuriant naują transakciją.

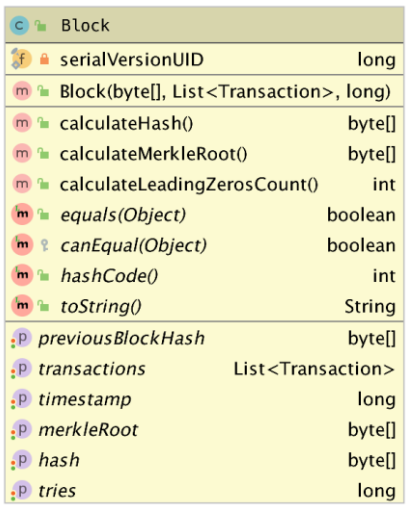
Grandinės bloko duomenys saugomi klasės *Block* objektuose (3.7 pav.). Šie objektai laiko informaciją apie patį bloką ir turi sąrašą konkrečių transakcijų, kurios buvo įtrauktos į bloką vykdant kasimo procesą. Duomenų laukai ir jų aprašymas pateikiami 3.3 lentelėje.

3.3 lentelė. Bloko struktūra

Duomenų laukas	Aprašymas
hash	Bloko maišos reikšmė
previousBlockHash	Praeito bloko maišos reikšmė
transactions	Transakcijų sąrašas
merkleRoot	*Merkle* šaknis
tries	Bandymų skaičius iki sėkmingo bloko iškasimo
timestamp	Laiko žyma

Kaip *Transaction*, taip ir *Block* klasė turi metodus tam tikriems duomenims apskaičiuoti. Žemiau pateikiamas šių metodų išeities kodas tokia metodų tvarka: bloko maišos reikšmės apskaičiavimas (*calculateHash*), *Merkle* šaknies apskaičiavimas (*calculateMerkleRoot*), maišos reikšmės pradinių nulių kiekio gavimas (*getLeadingZerosCount*):

```
public byte[] calculateHash() {
    byte[] hashableData = ArrayUtils.addAll(previousBlockHash, merkleRoot);
    hashableData = ArrayUtils.addAll(hashableData, Longs.toByteArray(tries));
    hashableData = ArrayUtils.addAll(hashableData, Longs.toByteArray(timestamp));
    return DigestUtils.sha256(hashableData);
}

public byte[] calculateMerkleRoot() {
    Queue<byte[]> hashQueue = transactions.stream()
            .map(Transaction::getHash)
            .collect(Collectors.toCollection(LinkedList::new));

    while (hashQueue.size() > 1) {
        byte[] hashableData = ArrayUtils.addAll(hashQueue.poll(), hashQueue.poll());
        hashQueue.add(DigestUtils.sha256(hashableData));
    }

    return hashQueue.poll();
}

public int getLeadingZerosCount() {
    for (int i = 0; i < getHash().length; i++) {
        if (getHash()[i] != 0) {
            return i;
        }
    }

    return getHash().length;
}
```



3.7 pav. Bloko klasė

3.4. Saugumo realizacija

Remiantis sistemos projektu buvo realizuotas sprendimas duomenims apsaugoti.

Visi sistemoje veikiantys mazgai, privalo turėti unikalią RSA raktų porą. Paleidimo metu atliekamas raktų patikrinimas. Jeigu jie jau egzistuoja diske – raktai įkraunami, validuojami ir sistema pasiruošia darbui. Jei raktai nerasti – automatiškai sukuriama nauja pora, išsaugoma naudotojo diske ir sistema gali tęsti darbą su ką tik sukurtais raktais. Prototipo realizacijoje panaudotas 2048 bitų ilgio RSA raktas su ECB šifravimo režimu ir PKCS1 užpildymu (angl. padding). RSA raktų pora sistemoje

veikia kaip naudotojo identifikatorius, o taip pat naudojama ir papildomų simetrinės kriptografijos raktų šifravimui, kurie skirti didelės apimties duomenims šifruoti.

Sistemoje realizuotas didelių duomenų šifravimas, kuriam RSA algoritmas nėra tinkamas, nes juo galima šifruoti ne didesnius duomenis nei pats rakto dydis. Šis šifravimas atliekamas naudojant simetrinę kriptografiją. Konkrečiai šio prototipo atveju pasirinkta naudoti AES raktą su CBC šifravimo režimu ir PKCS5 užpildymu. Tokio tipo raktai naudojami visų transakcijų turiniui ir naudotojo įkeliamiems failams šifruoti. Kiekvienai naujai transakcijai sukuriamas naujas AES raktas, kuriuo užšifruojamas visas pagrindinis turinys. Transakcijoje išsaugoma tiek rakto kopijų, kiek naudotojų galės peržiūrėti transakcijos turinį. Kiekviena AES rakto kopija transakcijoje užšifruojama konkretaus naudotojo viešuoju RSA raktu. Norint perskaityti transakcijos turinį, reikia iššifruoti turinio raktą su RSA privačiuoju raktu ir panaudoti gautą AES raktą turiniui iššifruoti. Šis mechanizmas užtikrina, kad transakcijos turinį perskaitys tik tie naudotojai, kurie turi reikiamą RSA raktą. Įprastu atveju, transakcijoje saugoma tik viena AES rakto kopija, nes turinį peržiūrėti gali tik tas naudotojas, kuris sukūrė transakciją. Kai atliekama failo dalinimosi operacija, transakcijoje išsaugoma papildoma AES rakto kopija, kuri skirta būtent tam naudotojui, su kuriuo norima pasidalinti failu.

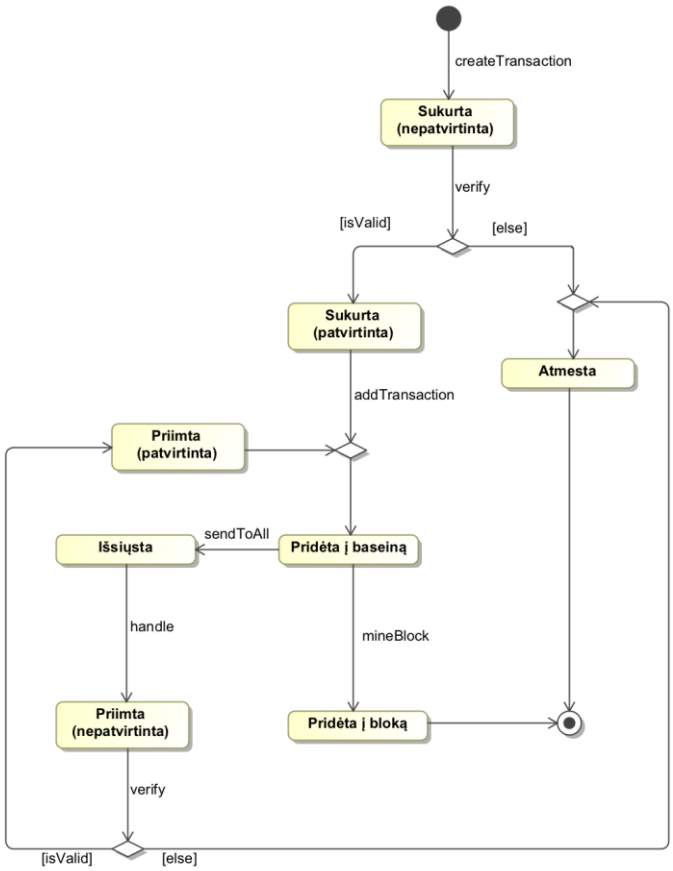
3.5. Sistemos būsenos

Sukurtos sistemos ir jos komponentų būsena gali kisti bet kuriuo veikimo momentu. Sistema paremta tinklo mazgų bendravimu ir duomenų mainais. Tai dinaminė aplinka, kurioje gali nutikti įvairios probleminės situacijos, kurias teks spręsti, siekiant sistemą sugrąžinti į stabilią būseną. Kiekvienas sistemos mazgas, atliekantis naujų blokų kasimą, varžosi tarpusavyje, kuris tai padarys greičiau. Kartais nutinka, kad sekantį bloką panašiu metu iškasa du mazgai ir pasidalina naujuoju bloku su kitais tinklo dalyviais. Mazgas, gavęs naują bloką, jį patikrina. Vienas iš kriterijų – turi būti teisinga praeito bloko maišos reikšmė. Tinkle pasklinda du skirtingi kandidatai į grandinę, kurie gali būti sekantys blokai, tačiau tik vienas gali būti išsaugotas ir įrašytas į grandinę. Kai sistema gauna bloką, kuris neatitinka apibrėžtų tikrinimo taisyklių, jis atmetamas ir sistema padidina savo stabilumo faktorių vienetu. Stabilumo faktorius – tai skaitliukas, kuris skaičiuoja, kiek netinkamų blokų gauta iš sistemos tinklo. Kai skaitliukas pasiekia reikšmę 3, mazgas skaitomas nestabiliu. Taigi, yra dvi pagrindinės sistemos būsenos:

- stabili;
- nestabili.

Perėjus į nestabilią būseną, sistema pradeda atsistatymo procesą. Šio proceso metu stabdoma naujų operacijų vykdymas kliento dalyje ir prašoma kitų tinklo mazgų pagalbos. Siunčiama užklausa į tinklą ir prašoma atsiųsti nesugadintą blokų grandinės versiją. Gavus blokų grandinę, atliekamas patikrinimas ir versijos išsaugojimas. Stabilumo faktorius atstatomas į pradinę reikšmę ir galima tęsti darbą su sistema.

Kita svarbi dalis susijusiu su būsenomis – tai transakcijos būsena. Per visą gyvavimo ciklą, transakcija pakeičia savo būseną ne vieną kartą. Kai transakcija sukuriama, ji tampa sukurta, tačiau nepatvirtinta. Toliau atliekamas jos duomenų patikrinimas. Jei yra klaidų – transakcija tampa atmesta ir toliau nėra apdorojama. Jei patikrinimas sėkmingas, transakcija tampa sukurta ir patvirtinta. Sekantis žingsnis – pridėti transakciją į neiškastų transakcijų baseiną, taigi būsena kinta dar kartą – pridėta į baseiną. Toliau vietinėje sistemoje gali būti atliekamas blokų kasimas ir, iškasus bloką, transakcija tampa pridėta į bloką. Po to, kai transakcija pridedama į baseiną, ja pasidalinama su kitais tinklo nariais – ji tampa išsiųsta. Mazge, kuris priima transakciją, pakartojamos jau prieš tai apibrėžtos būsenos, kuomet transakcija yra nepatvirtinta, atliekamas patikrinimas ir ji tampa patvirtinta. Toliau seka pridėjimas į baseiną ir bloko kasimas. Sėkmingu atveju, galutinė būsena transakcijai visais atvejais – pridėta į iškastą bloką. Transakcijos būsenų diagrama pavaizduota 3.8 paveiksle.



3.8 pav. Transakcijos būsenų diagrama

3.5.1. Diegimas ir paleidimas

Prototipo diegimas ir paleidimas padarytas taip, kad pradėti darbą su sistema būtų kuo paprasčiau. Kūrimui buvo panaudota Java programavimo kalba, taigi visas sukompiliuotas kodas patogiai supakuojamas į vieną *bc-filesystem.jar* bylą. Paleidimo metu reikia turėti pastarąją bylą ir sistemos nustatymų failą *application.properties* viename kataloge. Taip pat kompiuteryje jau turi būti įdiegta JRE (Java Runtime Environment) aplinka. Paleidimas atliekamas įvykdant komandą komandinėje eilutėje:

```
java -jar bc-filesystem.jar edu.ktu.blockchainfs.NodeRunner
```

Papildomai galima perrašyti bet kurį programos nustatymą iš *application.properties*, nekeičiant paties failo, pridedant papildomą parametrą paleidimo komandoje. Pavyzdžiui, norint pakeisti programos prievadą, galima pridėti parametrą *node.port* su reikšme *8334*. Tuomet komanda atrodytų taip:

```
java -jar bc-filesystem.jar edu.ktu.blockchainfs.NodeRunner —node.port=8334
```

Programos vykdymą ir atliekamus veiksmus galima stebėti įvykių žurnale, kurio pavyzdys pateikiamas 3.9 paveiksle. Čia vaizduojamas sistemos mazgo startavimas, raktų poros parinkimas ir patikrinimas, blokų kasimo mechanizmo paleidimas, prisijungimų klausymasis, ryšio užmezgimas su kitu mazgu, blokų grandinės atsiuntimo užklausimas ir įkrovimas, naujos transakcijos sukūrimas, bloko kasimas ir jo pridėjimas į blokų grandinę.

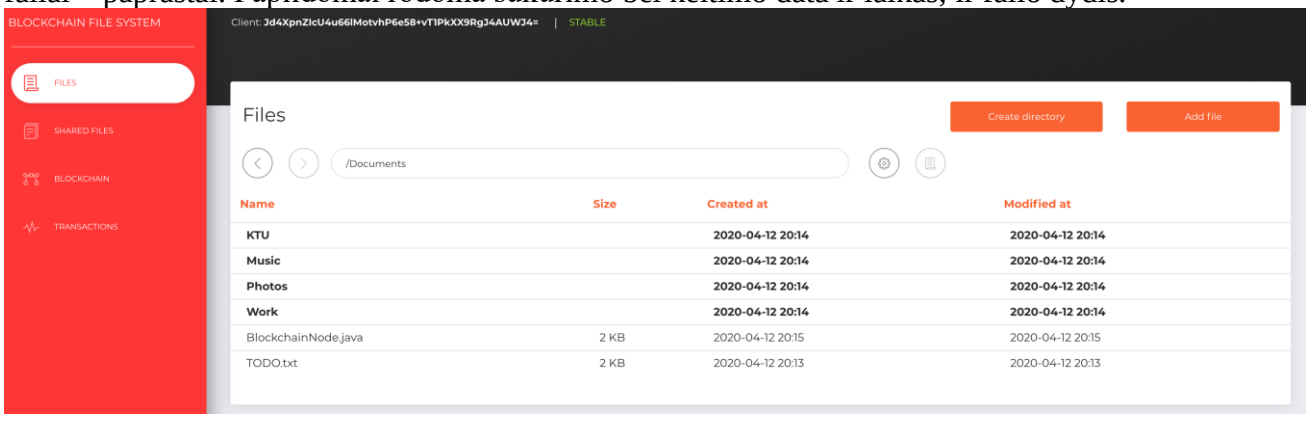
```
19:07:30 : INFO  : StartupInfoLogger         : Started NodeRunner in 4.008 seconds (JVM running for 4.773)
19:07:30 : INFO  : NodeSocketFactory         : Starting server socket on port 9333
19:07:30 : INFO  : NodeSocketFactory         : Server socket started
19:07:30 : INFO  : NodeKeysManager           : User key pair location:
Private key: /Users/tautvydastraska/Documents/Work/KTU/MBD/git/mbd_prototipas/key.priv
Public key: /Users/tautvydastraska/Documents/Work/KTU/MBD/git/mbd_prototipas/key.pub
19:07:30 : INFO  : NodeKeysManager           : Key pair verified successfully
19:07:30 : INFO  : BlockchainRequesterTask   : Waiting to request for blockchain
19:07:30 : INFO  : MinerTask                 : Miner started
19:07:30 : INFO  : ConnectionListenerTask    : Listening for connections...
19:07:32 : INFO  : BlockchainRequesterTask   : Waiting to request for blockchain
19:07:34 : INFO  : BlockchainRequesterTask   : Waiting to request for blockchain
19:07:35 : INFO  : LifeTickerTask            : Initializing connections with master nodes...
19:07:35 : INFO  : NodesManager              : Adding new node: Node(host=127.0.0.1, port=8333)
19:07:35 : INFO  : NodesManager              : Nodes: 127.0.0.1:8333;
19:07:35 : INFO  : ConcurrentConnector       : Initializing connection to 127.0.0.1:8333
19:07:35 : INFO  : ConcurrentConnector       : Connected to 127.0.0.1:8333
19:07:35 : INFO  : ConcurrentConnector       : Introduction sent to 127.0.0.1:8333
19:07:35 : INFO  : ConnectionHandler         : Connection accepted [127.0.0.1:8333]
19:07:35 : INFO  : SignalHandler             : Approval received from Node(host=127.0.0.1, port=8333). Requesting for known nodes...
19:07:36 : INFO  : BlockchainRequesterTask   : Waiting to request for blockchain
19:07:38 : INFO  : BlockchainRequesterTask   : Requesting for blockchain from Node(host=127.0.0.1, port=8333)
19:07:38 : INFO  : BlockchainTransferHandler : Initial blockchain received from Node(host=127.0.0.1, port=8333)
19:07:38 : INFO  : BlockchainManager         : Loading blockchain with size 1
19:07:38 : INFO  : QueryTranslatorFactoryInitiator : HHH000397: Using ASTQueryTranslatorFactory
19:07:38 : INFO  : TransactionFactory        : New transaction created
19:07:38 : INFO  : TransactionManager        : New tx [txps: 1]
19:07:40 : INFO  : MinerTask                 : Mined block [txc: 1, nonce 105107]
19:07:40 : INFO  : BlockchainManager         : Block appended [bcs: 2, txps: 0]
```

3.9 pav. Veikiančio mazgo įvykių žurnalo iškarpa

3.5.2. Sistemos vaizdas

Failų sistema pasiekiama per grafinę naudotojo sąsają naršyklėje. Sistemoje realizuoti pagrindinį funkcionalumą įgalinantys langai, atitinkantys pradinę rekomendaciją projekto apraše (2.1.3 Rekomenduojama naudotojo sąsaja). Kairėje matomas meniu, kurio pagalba galima naviguoti tarp failų ar blokų grandinės peržiūros. Viršuje pateikiamas kliento kodas, ant kurio paspaudus jis automatiškai nukopijuojamas į iškarpinę (angl. clipboard) – tokiu būdu juo lengva pasidalinti su kitu žmogumi, jei norime, kad su mumis galėtų pasidalinti failu. Šalia kliento kodo pateikiama sistemos būsena. Jei viskas gerai – matomas žalias užrašas „STABLE“. Atsiradus ryšio problemoms su tinklu arba jeigu turima vietinė blokų grandinės kopija sugadinta – rodomas raudonas užrašas „UNSTABLE“ tol, kol sistema išsprendžia problemą.

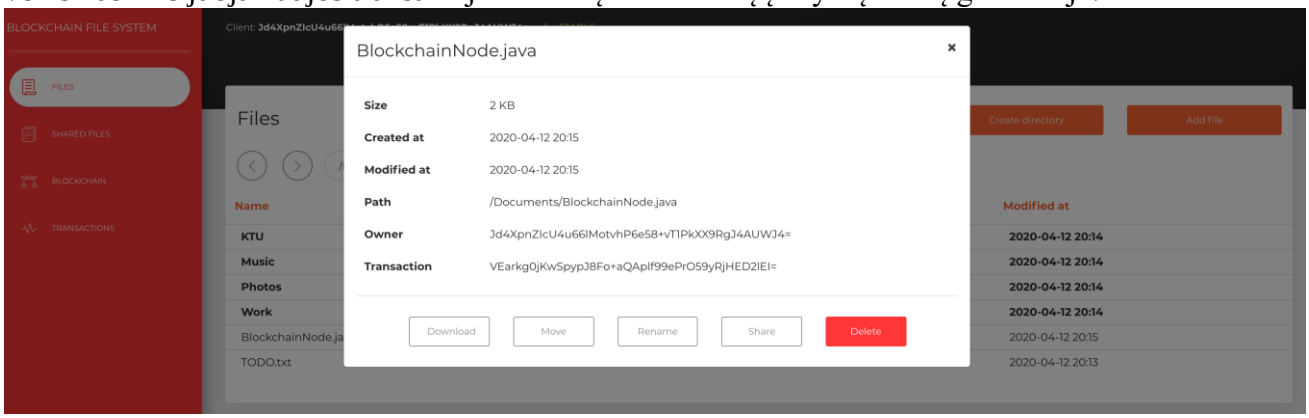
Failų naršymo lange (3.10 pav.) galima peržiūrėti įkeltus failus ir sukurtus katalogus bei atlikti tam tikras funkcijas – peržiūrėti failo detales, naviguoti kataloguose, atverti katalogą pagal failų sistemos kelią, kurti katalogą ar įkelti naują failą. Katalogai sąraše vaizduojami paryškintu šriftu, o failai – paprastai. Papildomai rodoma sukūrimo bei keitimo data ir laikas, ir failo dydis.



3.10 pav. Failų naršymo langas

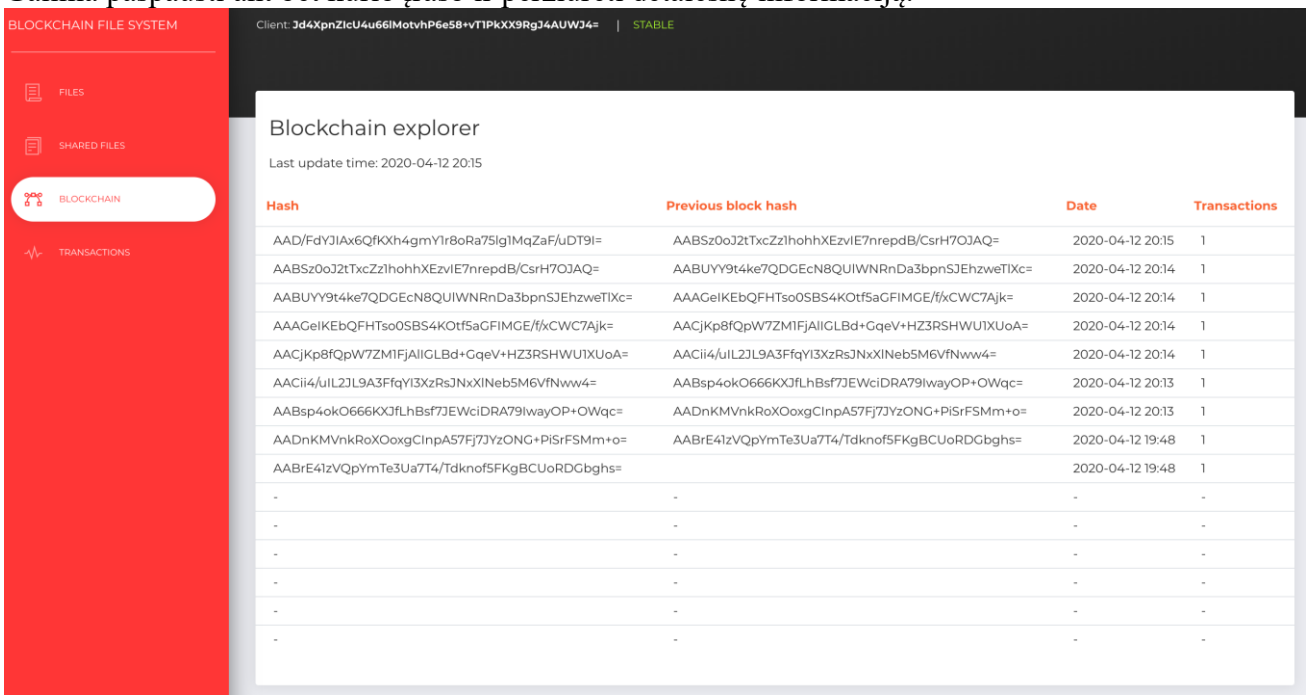
Pasirinkus ir paspaudus ant bet kurio failo, atidaromas failo detalių langas (3.11 pav.), kuriame taip pat galima pasirinkti norimą atlikti operaciją su šiuo konkrečiu failu. Čia rodomas failo dydis, sukūrimo data, keitimo data, pilnas kelias iki failo, savininko kliento kodas ir transakcijos, kurioje įrašyta failo informacija, maišos reikšmė. Operacijos, kurias galima pasirinkti failui: atsiųsti,

perkelti į kitą katalogą, pervardinti, pasidalinti su kitu naudotoju ir ištrinti iš failų sistemos. Kiekvienas veiksmas inicijuoja naujos transakcijos kūrimą ir duomenų įrašymą blokų grandinėje.



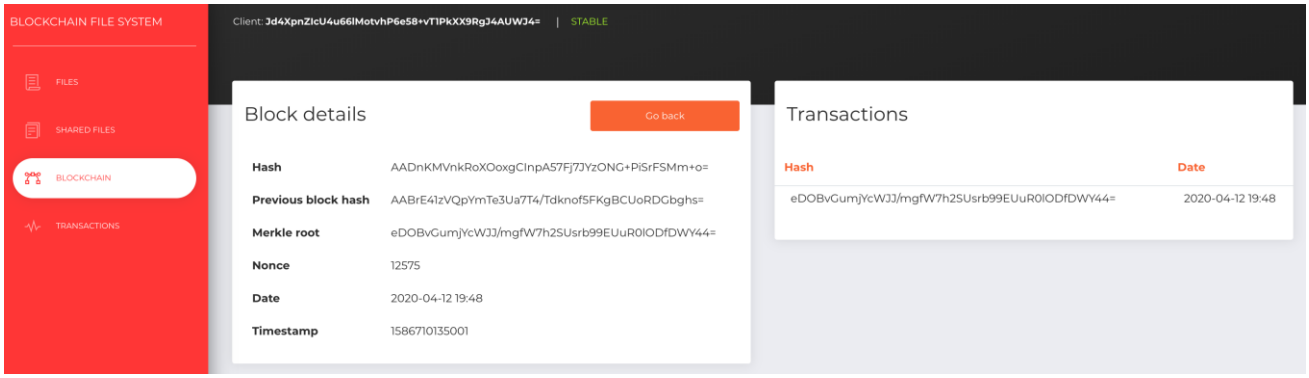
3.11 pav. Failo detalės ir operacijos

Meniu pasirinkus „BLOCKCHAIN“, atveriamas blokų grandinės peržiūros langas, su pilna blokų grandine (3.12 pav.). Rodoma kiekvieno bloko maišos reikšmė, nuoroda į ankstesnį bloką, sukūrimo data ir bloke įrašytų transakcijų skaičius. Šiame sąraše puikiai matomas grandinės nuoseklumas ir kiekvieno bloko ryšys sudarant grandinę. Viršuje rodomas pats naujausias blokas, o pirmasis blokas apačioje neturi nuorodos į ankstesnį bloką, nes nuo jo prasideda blokų grandinė. Galima paspausti ant bet kurio įrašo ir peržiūrėti detalesnę informaciją.



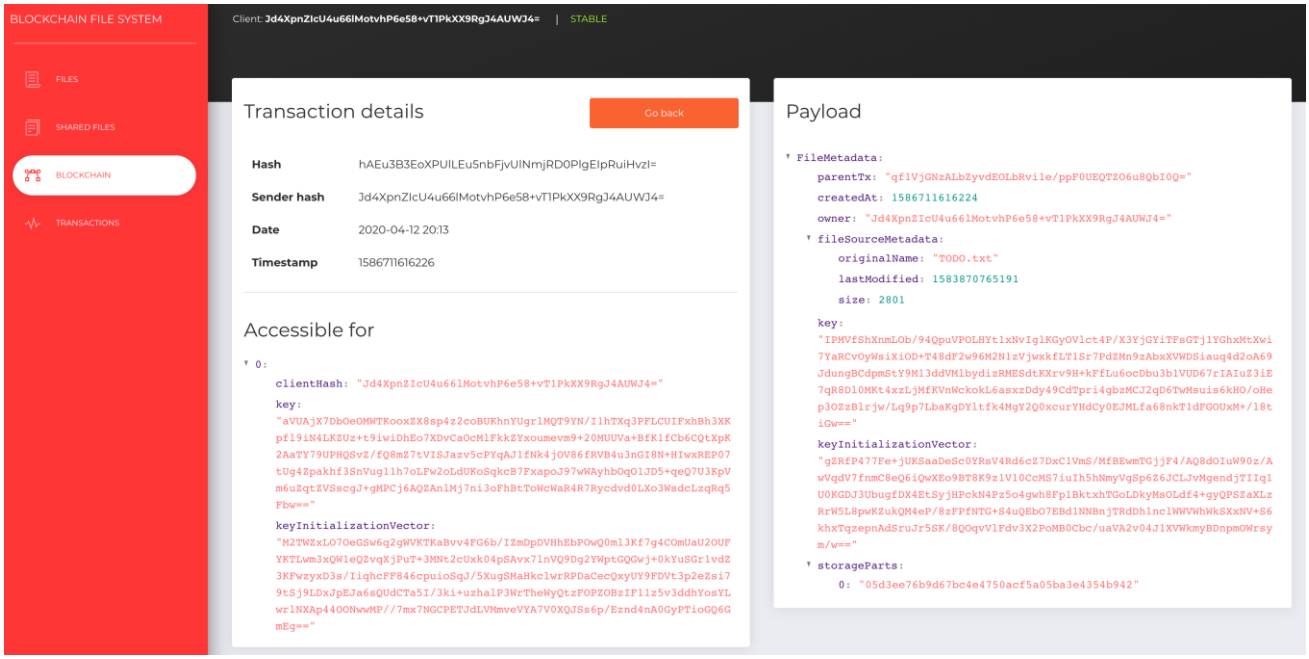
3.12 pav. Blokų grandinės naršymo langas

Pasirinkus bloką iš sąrašo, atveriamas detalių peržiūros langas (3.13 pav.). Čia pateikiama daugiau informacijos apie bloką: maišos reikšmė, ankstesnio bloko maišos reikšmė, *Merkle* šaknies reikšmė, *nonce* reikšmė, data ir laikas bei laiko žyma. Dešinėje rodomos transakcijos, kurios pateko į šį bloką. Transakciją galima pasirinkti ir peržiūrėti detalesnę informaciją.



3.13 pav. Bloko peržiūros langas

Transakcijos peržiūros langą (3.14 pav.) sudaro tokia informacija: transakcijos maišos reikšmė, siuntėjo kliento kodas, data ir laikas bei laiko žyma. Žemiau rodoma informacija apie klientą, kuris gali peržiūrėti transakcijos turinį. Kartu pateikiamas ir užšifruotas raktas turiniui atrakinti. Dešinėje vaizduojamas pagrindinis transakcijos turinys. Šiame pavyzdyje – tai naujo failo įkėlimo transakcija. Matoma informacija apie failą, saugykloje saugomų failo dalių nuorodos ir raktas failui iššifruoti.



3.14 pav. Transakcijos peržiūros langas

3.6. Išvados

Realizuotas failų sistemos prototipas pagal antrame skyriuje aprašytą projektą. Realizacijoje pritaikyta blokų grandinių technologiją, kurios dėka sistemai suteikiamos šios teigiamos savybės:

- sutrikdžius vieną mazgą, lieka daug grandinės kopijų visame tinkle, taigi informacija neprarandama ir ją lengva parsiųsti iš naujo;
- praktiškai neįmanomas duomenų klastojimas, nes transakcijos ir blokai validuojami daugelio mazgų pagalba visame tinkle;
- validacija daugelyje mazgų užtikrina sistemos ir duomenų vientisumą;
- tai paskirstyta sistema, taigi užtikrinamas pasiekiamumas;
- sistema decentralizuota, nėra vieno tiesos šaltinio, sprendimai priimami autonomiškai tarp visų tinkle veikiančių mazgų;
- paskirstyta sistema užtikrina pasiekiamumą;
- konfidencialumas užtikrinamas pasitelkiant papildomas kriptografines priemones šifruojant failus ir failų metaduomenis.

Aprašyta prototipo architektūra ir realizuoti komponentai. Sistemoje įgalintos visos failų sistemos operacijos, kurios buvo numatytos projekto (metodo siūlymo) dalyje. Saugumo realizacijos dalyje aprašytas konkrečių kriptografinių algoritmų parinkimas, o pats procesas apsaugant failus ir transakcijas realizuotas pagal sistemos projekto aprašą. Išspręsta blokų grandinės neefektyvumo problema esant poreikiui greitai gauti grandinėje saugomus duomenis. Tai įgyvendinta panaudojant atmintyje saugomą reliacinę duomenų bazę, kuri vietiniame mazge nuolatos sinchronizuojama su blokų grandine. Aprašytos galimos sistemos ir transakcijų būsenos. Pateiktos instrukcijos sistemai paleisti ir supažindinama su failų sistemos grafine kliento naudotojo sąsaja.

4. REALIZUOTOS FAILŲ SISTEMOS TYRIMAS

Realizuotą failų sistemos prototipą pasirinkta ištirti dvejais būdais: kokybiškai ir kiekybiškai. Kokybinis tyrimas atliekamas palyginant centralizuotą informacijos saugojimo metodą su sukurtu prototipu pagal pasirinktas charakteristikas. Taip pat išskiriamos teigiamos ir neigiamos sukurtos sistemos savybės. Kiekybinis tyrimas atliekamas tiesiogiai išbandant sistemą ir matuojant laiko be atminties sąnaudas, kurių reikalauja failų sistema. Pagal gautus rezultatus pateikiamos galutinės išvados sukurtai vystomo metodo realizacijai.

Tyrime naudojami matavimo vienetai:

- laikas – milisekundės (ms);
- atminties kiekis – baitai (B).

4.1. Kokybinis tyrimas

Kokybinio tyrimo metu palyginamos tradicinės centralizuotos duomenų bazės savybės ir blokų grandinės savybės, kurios yra taikomos ir sukurto prototipo realizacijai. Taip pat pateikiami blokų grandine paremtos failų sistemos pranašumai ir naudos bei trūkumai.

Blokų grandinės palyginimas su centralizuota duomenų baze pateikimas 4.1 lentelėje.

4.1 lentelė. Blokų grandinės palyginimas su centralizuota duomenų baze

Savybė	Centralizuota duomenų bazė	Blokų grandinė
Duomenų kontrolė	Centralizuota	Decentralizuota
Kas gali naudotis?	Tik autorizuotos šalys	Visi, jei pasirenkamas viešas blokų grandinės tipas
Architektūra	Kliento-serverio	Mazgas su mazgu (angl. peer-to-peer)
Duomenų pastovumas ir keičiamumas	Duomenis galima keisti bet kada, turint reikiamą prieigos teisę	Įrašyti duomenys nekeičiami
Istorinių duomenų peržiūra	Negalima be papildomo funkcionalumo pagalbos	Galima
Veikimo sparta	Greita	Sąlyginai lėta, priklausomai nuo tinkle veikiančių mazgų skaičiaus

Tyrimo ir bandymų metu pastebėtos ir išskirtos teigiamos ir neigiamos realizuoto failų sistemos prototipo savybės, turinčios įtakos produkto kokybei. Blokų grandine paremtos failų sistemos pranašumai ir nauda:

- sutrikdžius vieną mazgą (ar nustojus jam veikti), lieka daug grandinės kopijų visame tinkle, taigi informacija neprarandama ir ją lengva parsiųsti iš naujo;
- sunkus (praktiškai neįmanomas) duomenų klastojimas, nes transakcijos ir blokai validuojami daugelio mazgų pagalba visame tinkle;
- validacija daugelyje mazgų užtikrina sistemos ir duomenų vientisumą;
- į grandinę įrašytų duomenų pakeisti negalima – tai užtikrina duomenų vientisumą;
- tai decentralizuota (paskirstyta) sistema, taigi užtikrinamas pasiekiamumas;
- sistema decentralizuota, nėra vieno tiesos šaltinio, sprendimai priimami autonomiškai tarp visų tinkle veikiančių mazgų;
- matoma visa įvykių istorija;
- konfidencialumas užtikrinamas elektroniniu parašu ir pasitelkiant papildomas kriptografines priemones šifruojant transakcijų bei failų turinį.

Blokų grandine paremtos failų sistemos trūkumai:

- sąlyginai didelis elektros energijos suvartojimas sistemos funkcionavimui palaikyti;
- nustojus veikti visiems sistemos mazgams, prarandami visi duomenys;

- į grandinę įrašyti duomenys yra nekeičiami – tai ir privalumas, ir trūkumas. Trūkumas todėl, kad reikia įgyvendinti papildomas priemones siekiant realizuoti tam tikras sistemos operacijas. Pavyzdžiui, negalima tiesiog ištrinti duomenų apie šalinamą failą, reikia rašyti papildomus duomenis, kurie pažymėtų failą kaip ištrintą;
- kartais veikimas gali tapti neefektyvus tikrinant naujus duomenis dideliame sistemos tinkle.

4.2. Kiekybinis tyrimas

Kiekybinis tyrimas atliktas vykdant failų sistemos operacijas skirtingo dydžio tinkluose. Pasirinkta tyrimą atlikti dviejų ir dešimties mazgų tinkluose. Kiekviena failų sistemos operacija buvo kartojama 50 kartų, taip buvo gauta aiški informacija apie sistemos veikimą atminties ir laiko sąnaudų atžvilgiu.

4.2.1. Tyrimo planas

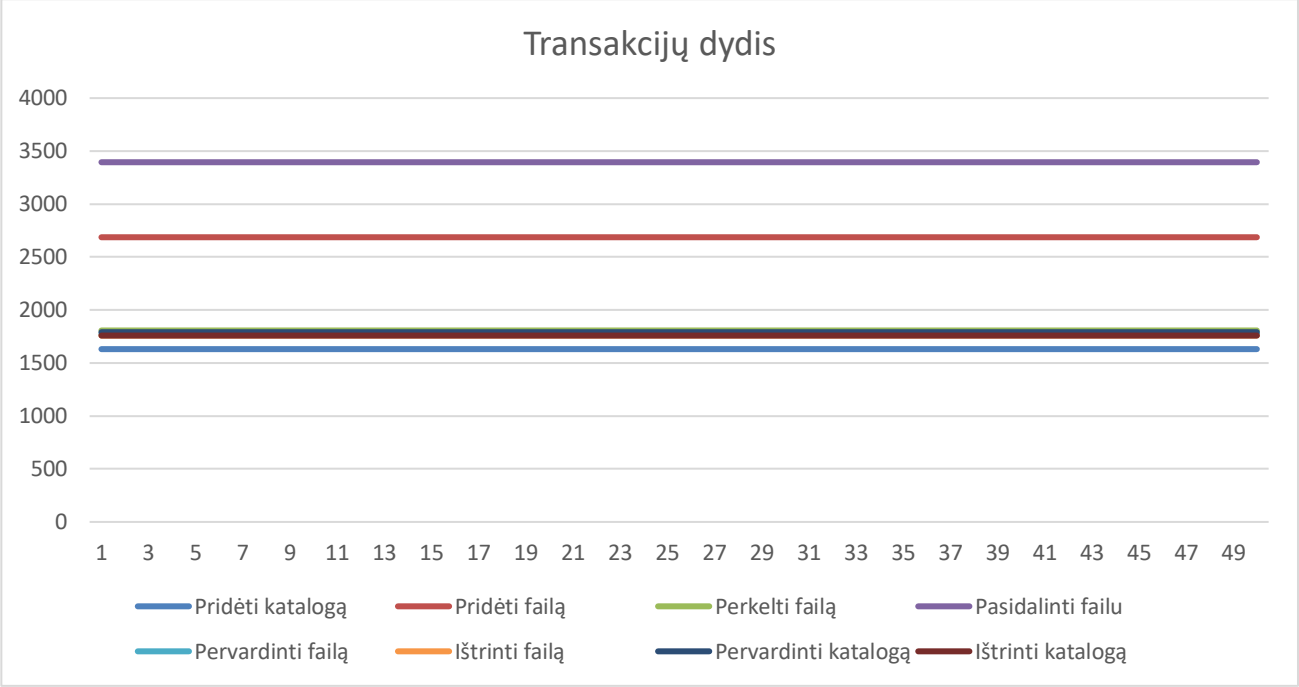
Parengtas tyrimo planas, kurio laikantis buvo surinkti metrikų duomenys ir grafiškai pavaizduoti diagramose. Planą sudaro šie etapai:

1. įvykdyti visas failų sistemos operacijas sistemoje, kurioje veikia du (2) mazgai;
   1.1. sukurti 50 naujų katalogų atsitiktiniais pavadinimais;
   1.2. išsaugoti 50 naujų failų, kurių dydis – 1 MB;
   1.3. atlikti 50 failų perkėlimo operacijų į atsitiktinai parenkamus katalogus;
   1.4. pasidalinti failais 50 kartų su kitu sistemos naudotoju;
   1.5. pervardinti 50 atsitiktinių failų atsitiktinai parinktais pavadinimais;
   1.6. ištrinti 50 failų;
   1.7. pervardinti 50 katalogų atsitiktinai parinktais pavadinimais;
   1.8. ištrinti 50 katalogų;
2. įvykdyti visas failų sistemos operacijas sistemoje, kurioje veikia dešimt (10) mazgų;
   2.1. sukurti 50 naujų katalogų atsitiktiniais pavadinimais;
   2.2. išsaugoti 50 naujų failų, kurių dydis – 1 MB;
   2.3. atlikti 50 failų perkėlimo operacijų į atsitiktinai parenkamus katalogus;
   2.4. pasidalinti failais 50 kartų su kitu sistemos naudotoju;
   2.5. pervardinti 50 atsitiktinių failų atsitiktinai parinktais pavadinimais;
   2.6. ištrinti 50 failų;
   2.7. pervardinti 50 katalogų atsitiktinai parinktais pavadinimais;
   2.8. ištrinti 50 katalogų;
3. surinkti atminties sąnaudų (transakcijų, blokų dydžio ir blokų grandinės augimo) duomenis;
4. surinkti laiko sąnaudų (transakcijų, blokų sukūrimo ir patvirtinimo laikai) duomenis;
5. aprašyti gautus rezultatus.

4.2.2. Atminties sąnaudos

Atminties sąnaudų tyrime pamatuotas sukurtų transakcijų, blokų ir grandinės dydis pagal įvykdytas failų sistemos operacijas. Visų pirma pateikiama informacija apie transakcijas (4.1 pav.). Pagal pateikiamą diagramą galima matyti, kad daugumos operacijų sukuriamos transakcijos užima apie 1,7 KB. Daugiausiai vietos reikalauja dviejų tipų transakcijos – pridedant failą (2,6 KB) ir dalinantis failu (3,4 KB). Taip yra todėl, kad transakcijoje saugomi papildomi AES kriptografiniai raktai, kurie buvo panaudoti failui šifruoti. Mažiausiai atminties reikalaujanti sistemos operacija – pridėti katalogą, kurios transakcija užima apie 1,6 KB. Testo metu, visų operacijų transakcijų dydžiai nekito. Įkeliant didesnį failą, būtų matomas transakcijos dydžio šuolis dėl didesnio kiekio nuorodų į failų dalis saugykloje.

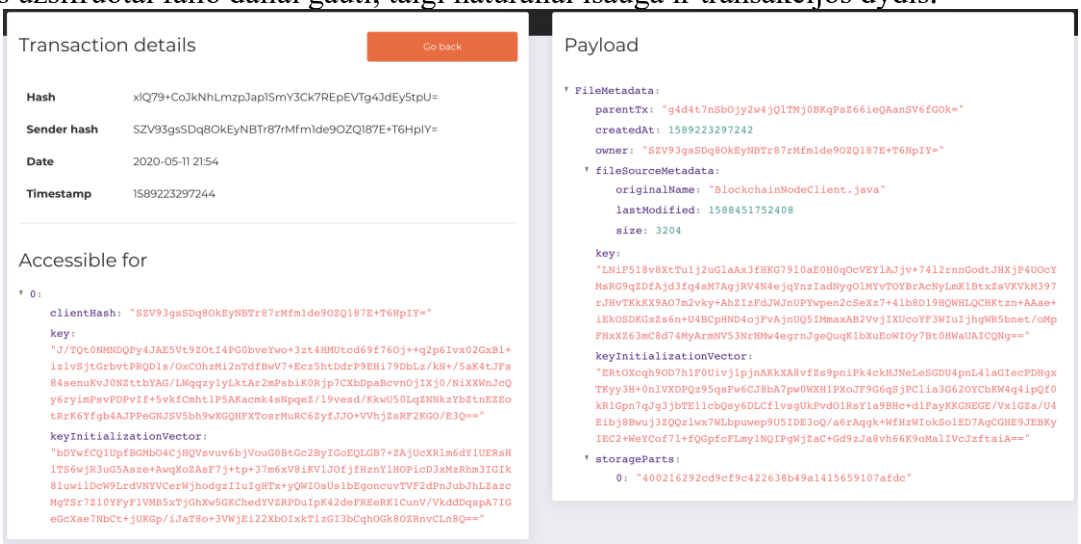
Atminties sąnaudų tyrimo duomenys pateikti priede 7.1.1 Transakcijų dydis.



4.1 pav. Transakcijos dydis pagal failų operacijas

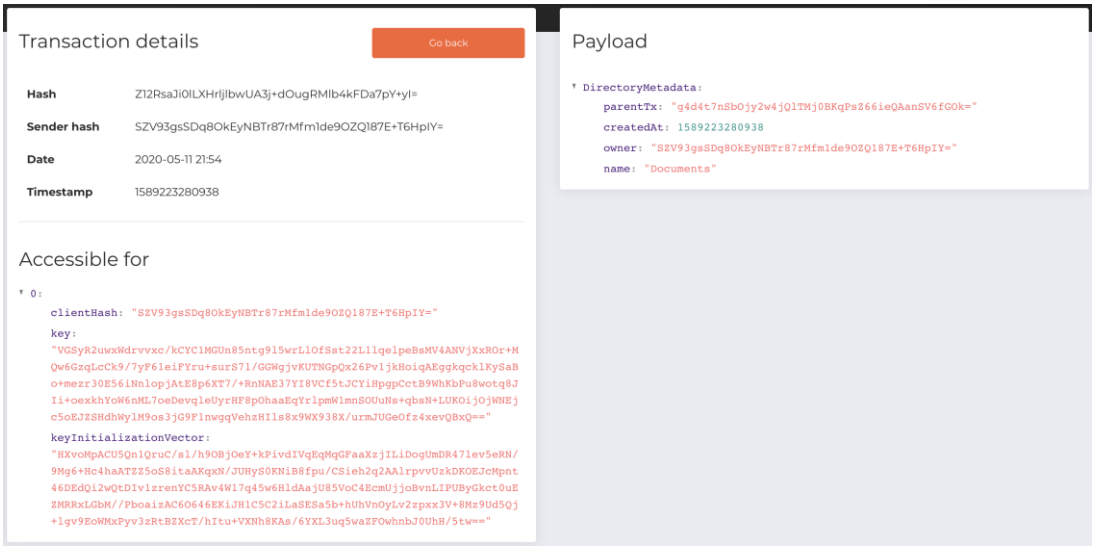
Toliau pateikiami transakcijų turinio vizualizacijos langai, pagal kurios galima matyti, kodėl atsiranda skirtumas tarp skirtingų operacijų transakcijų.

Naujo failo įkėlimo transakcijoje (4.2 pav.) išsaugotas papildomas raktas ir jo inicijavimo vektorius užšifruotai failo daliai gauti, taigi natūraliai išauga ir transakcijos dydis.



4.2 pav. Failo įkėlimo transakcijos pavyzdys

Katalogo sukūrimo transakcijoje (4.3 pav.) saugoma mažiau informacijos, nėra papildomo šifravimo rakto, taigi reikia mažiau atminties siekiant įvykdyti šią operaciją.

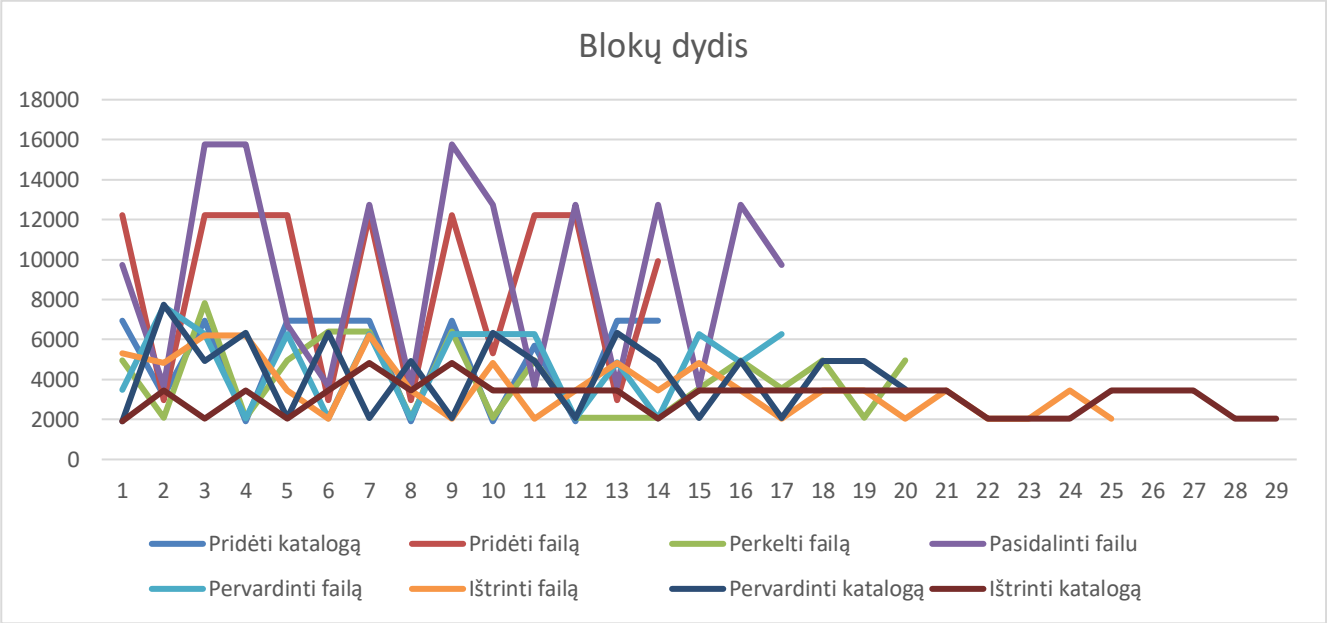


4.3 pav. Katalogo sukūrimo transakcijos pavyzdys

Blokų dydis tiesiogiai priklauso nuo transakcijų dydžio, kurios patenka į juos. Testo metu į viena bloką buvo leidžiama išsaugoti ne daugiau nei penkias transakcijas. Diagramoje pavaizduotas naujų blokų dydžio svyravimas (4.4 pav.). Taip yra todėl, kad ne vienodas kiekis transakcijų patenka į kiekvieną bloką. Dėl šios priežasties, blokų sukurta mažiau nei transakcijų. Be transakcijų, prie bloko dydžio prisideda ir jo duomenys – maišos reikšmė, praeito bloko maišos reikšmė, *Merkle* šaknis, *nonce* ir laiko žymas. Visas šis paketas sudaro bloko dydį.

Iš gautų duomenų galima matyti, kad atminties sąnaudų atžvilgiu dominuoja tos pačios operacijos – pridėti failą ir pasidalinti failu. Dėl šių operacijų, vieno bloko dydis šoktelėjo iki 16 KB. Vidutiniškai vienas blokas užima apie 5 KB.

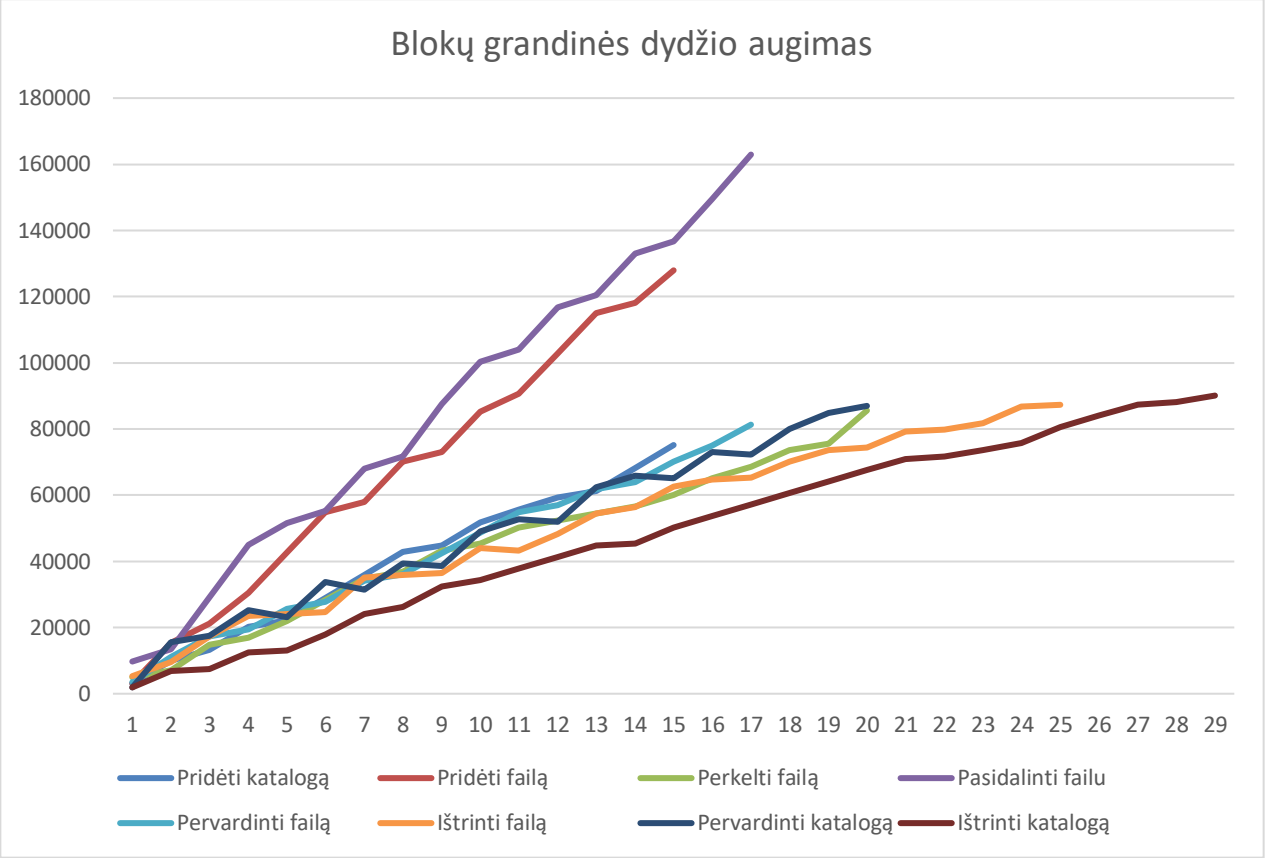
Blokų dydžio tyrimo duomenys pateikti priede 7.1.2 Blokų dydis.



4.4 pav. Blokų dydžio priklausomybė nuo įvykdytų operacijų

Toliau pateikiamas blokų grandinės augimas pagal įvykdytas failų sistemos operacijas (4.5 pav.). Sparčiausias augimas įvyko išsaugant naujus failus ir dalinantis jais su kitais naudotojais. Po 50-ies pasidalinimų failais, grandinė išaugo iki 160 KB dydžio. Dauguma operacijų po 50 pakartojimų nesiekė 100 KB blokų grandinės dydžio ribos. Tai vienos lokalios blokų grandinės kopijos dydis. Kiekvienas mazgas turi atskirą kopiją, taigi bendras saugomų grandinės duomenų dydis priklauso nuo to, kiek sistemoje vienu metu veikia mazgų.

Blokų grandinės dydžio augimo tyrimo duomenys pateikti priede 7.1.3 Blokų grandinės augimas.



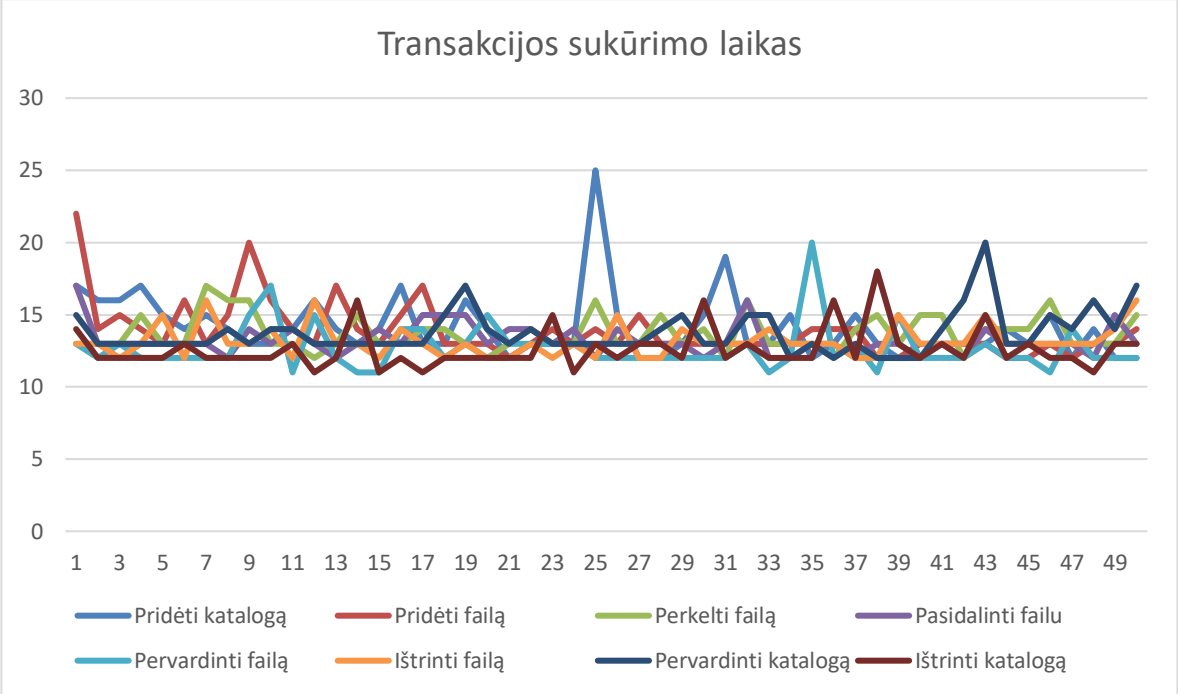
4.5 pav. Blokų grandinės augimo priklausomybė nuo įvykdytų operacijų

4.2.3. Laiko sąnaudos

Tyrimui iškeltą uždavinį sudaro ir laiko sąnaudų tyrimas – reikia išsiaiškinti, kiek laiko trunka sukurti ir patvirtinti transakcijas bei kiek laiko trunka sukurti naują bloką ir patvirtinti jį tinkle prijungiant naująjį bloką prie blokų grandinės. Tyrimo metu buvo laikomasi sudaryto plano – pamatuotos laiko sąnaudos vykdant kiekvieną failų sistemos operaciją penkiasdešimt kartų dviejų ir dešimties mazgų dydžio sistemoje (4.6 pav.).

Bet kokios operacijos inicijavimas ir transakcijos sukūrimas nėra labai laikui imlus procesas. Vidutiniškai, nepriklausomai nuo operacijos tipo, vieną transakciją sukurti užtrunka apie 13-14 ms. Laikas gali svyruoti dėl mazgo apkrautumo ir kūrimo metu apdorojamo duomenų kiekio, tačiau bet kokiu atveju, veiksmas atliekamas pakankamai greitai. Diagramoje galima matyti, kad kai kurių transakcijų kūrimo laikas šoktelėjo iki 20-25 ms. Bet kokiu atveju, rezultatas geras ir galima teigti, kad transakcijos sukuriamos greitai.

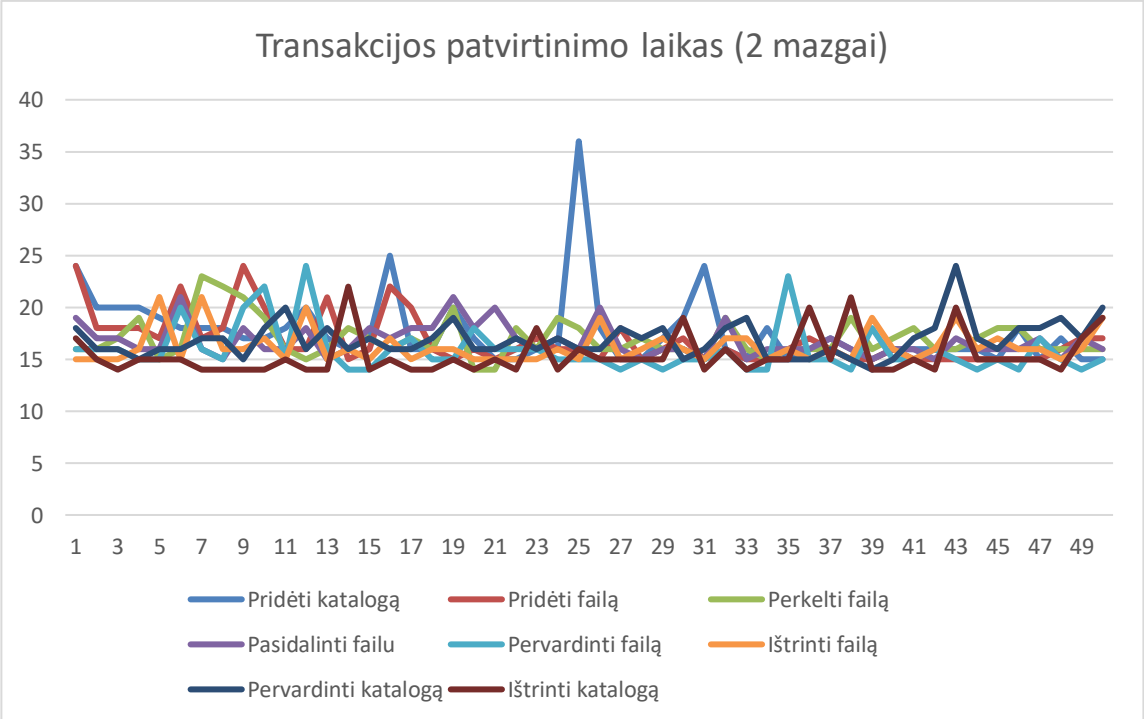
Transakcijos sukūrimo laiko tyrimo duomenys pateikti priede 7.1.4 Transakcijų sukūrimo laikas.



4.6 pav. Transakcijos sukūrimo laikas pagal failų operacijas

Transakcijos patvirtinimo laikas tinkle, kuriame veikia du mazgai, trunka panašiai tiek pat laiko (4.7 pav.), kiek ir sukūrimas. Vidutiniškai tai truko 16,5 ms. Transakcija laikoma patvirtinta, kai kiekvienas mazgas turi jos kopiją, sėkmingai atliko patikrinimą ir išsaugojo naujų transakcijų baseine. Prie gerų rezultatų prisidėjo ir tai, kad tyrimas buvo atliekamas tinkle, kuris veikia viename kompiuteryje. Galima tikėtis prastesnių rezultatų, jei mazgai veiktų nutolusiose mašinose.

Transakcijos patvirtinimo laiko dviejų mazgų sistemoje tyrimo duomenys pateikti priede 7.1.5 Transakcijų patvirtinimo laikas (2 mazgai).

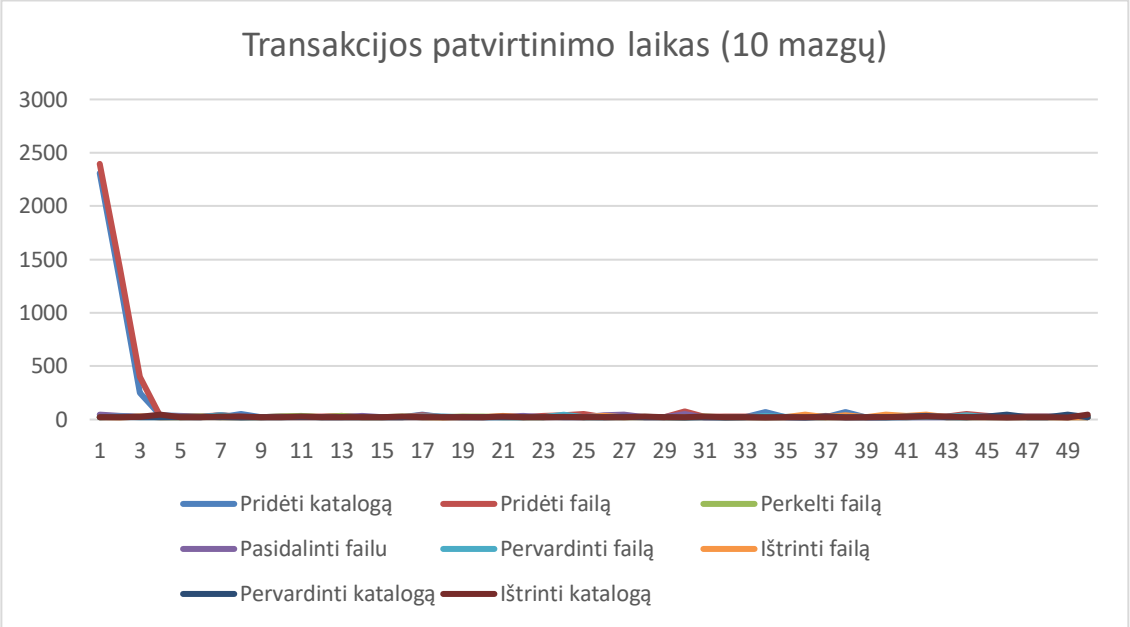


4.7 pav. Transakcijos patvirtinimo laikas dviejų mazgų sistemoje

Transakcijos patvirtinimo laikas dešimties mazgų sistemoje vaizduojamas dvejomis diagramomis. Pirmoji diagrama (4.8 pav.) apima visas transakcijas, o antroji (4.9 pav.) visas, išskyrus pirmąsias tris. Taip yra todėl, kad pirmosioms transakcijoms patvirtinti prireikė daug daugiau laiko nei

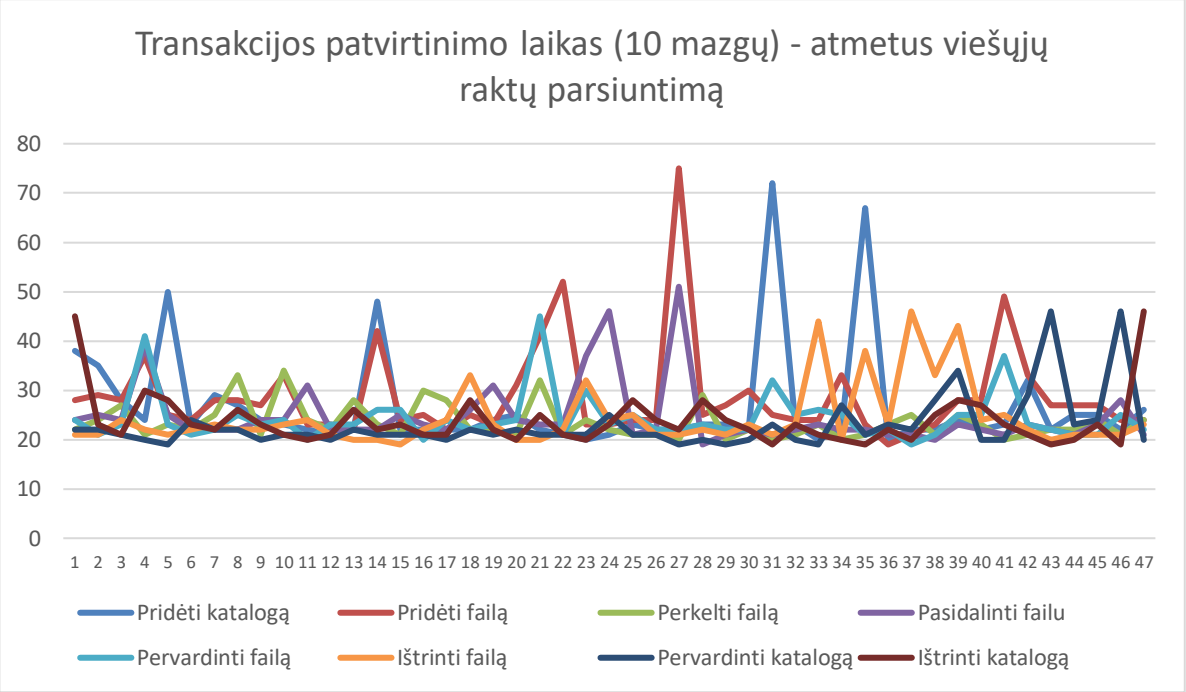
likusioms ir sunku matyti likusių transakcijų rezultatus. Transakcijų tikrinimui reikia gauti kūrėjo viešąjį raktą ir, jei raktas nėra žinomas, mazgas siunčia užklausą į tinklą raktui gauti. Pirmasis mazgas, kuris gali patenkinti užklausą, grąžina reikiamą raktą ir transakcijos patikrinimas gali būti tęsiamas. Dėl šio proceso, pirmųjų transakcijų tikrinimas užtruko ilgiau nei 2 sekundės, kol buvo gauti visi reikiami raktai. Vidutiniškai, viena transakcija buvo patvirtina per 44,7 ms. Taigi, galima teigti, kad patvirtinimo laikas priklauso nuo mazgų kiekio sistemoje.

Transakcijos patvirtinimo laiko dešimties mazgų sistemoje tyrimo duomenys pateikti priede 7.1.6 Transakcijų patvirtinimo laikas (10 mazgų).



4.8 pav. Transakcijos patvirtinimo laikas dešimties mazgų sistemoje

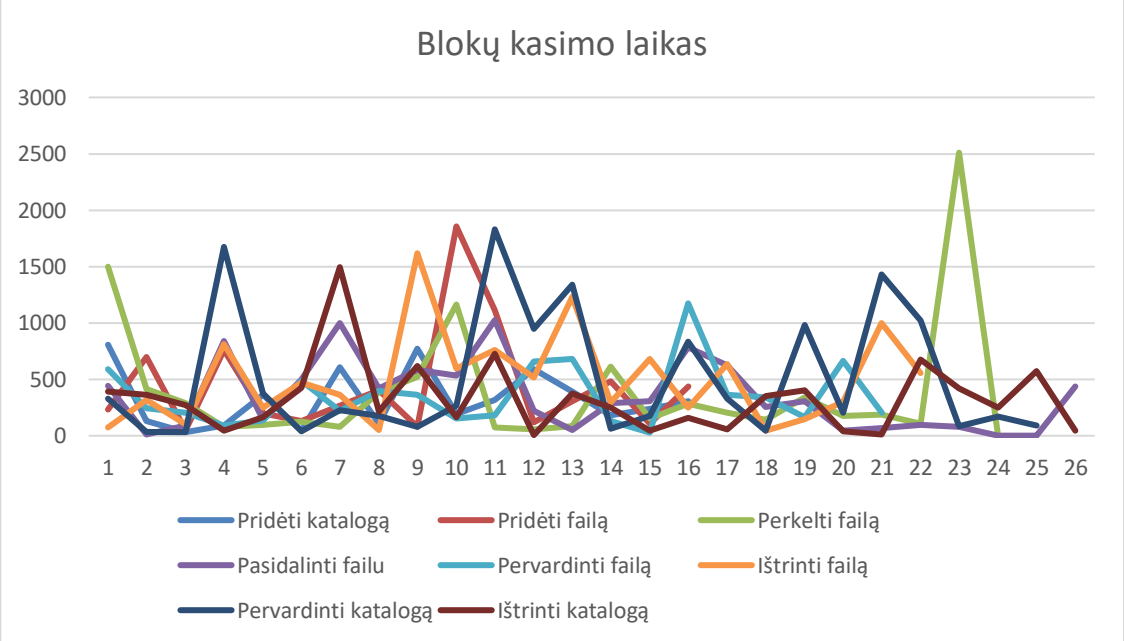
Atmetus pirmąsias transakcijas, galima aiškiau matyti, kiek trunka transakcijos patvirtinimas tinkle, kuriame veikia dešimt mazgų (4.9 pav.). Lyginant su dviejų mazgų sistema, skirtumas akivaizdus ir galima teigti, kad kuo daugiau mazgų, tuo ilgiau trunka atlikti vieną operaciją tokio tipo failų sistemoje. Bet kuriuo atveju, laikas neblogas – atmetus pirmąsias transakcijas, vidutiniškai viena patvirtinama per 25 ms.



4.9 pav. Transakcijos patvirtinimo laikas dešimties mazgų sistemoje (be pirmųjų transakcijų)

Toliau buvo matuojamas blokų kasimo laikas, kartojant kiekvieną failų sistemos operaciją penkiasdešimt kartų (4.10 pav.). Diagramose vaizduojama mažiau blokų nei buvo atlikta operacijų, nes į vieną bloką buvo paimta daugiau nei viena transakcija, atitinkanti vieną operaciją (ne daugiau penkios transakcijos viename bloke). Testo metu parinktas antro lygio blokų kasimo sudėtingumas. Tai reiškia, kad naujo bloko maišos reikšmė prasideda mažiausiai dviem nuliais. Diagramos duomenys surinkti iš tinklo su dešimt mazgų. Vidutiniškai vieno bloko kasimas truko apie 400 ms. Kai kuriais atvejais laikas išaugo iki 2 sekundžių ir kartą iki 2,5 sekundės. Kasimo greitis nepriklauso nuo transakcijų tipo, jų dydžio ar bloko dydžio. Didžiausią įtaką turi blokų kasimo sudėtingumas. Kuo sudėtingesnė matematinė užduotis, tuo ilgiau trunka iškasti naują bloką.

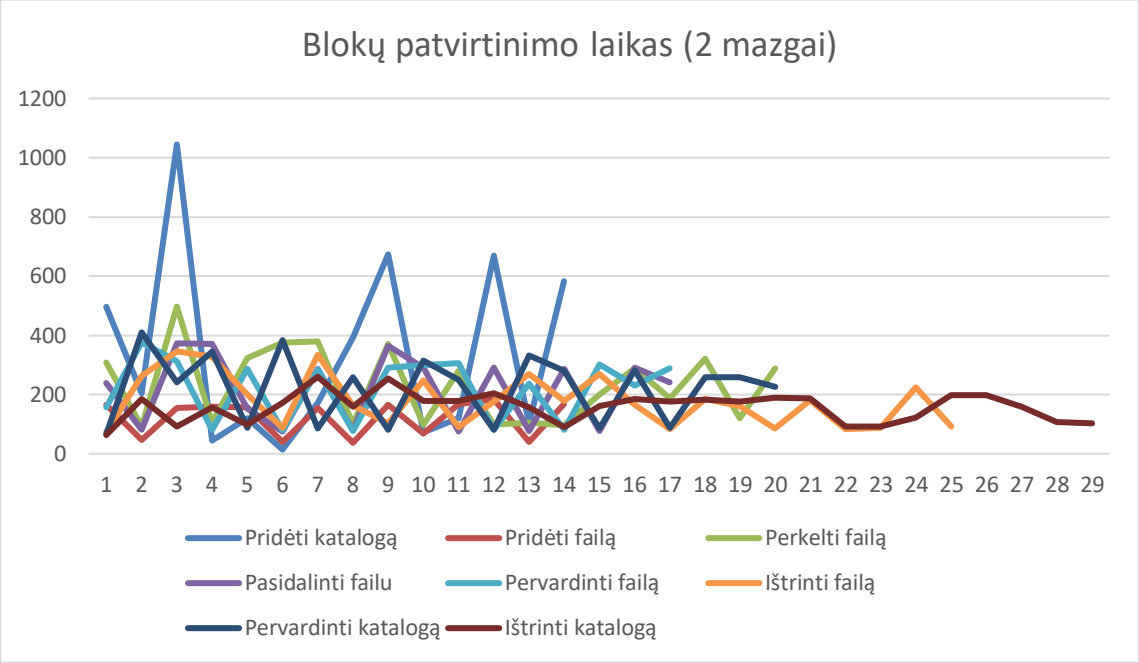
Blokų kasimo laiko tyrimo duomenys pateikti priede 7.1.7 Blokų kasimos laikas.



4.10 pav. Blokų kasimo laikas pagal failų sistemos operacijas

Naujų blokų patvirtinimas dviejų mazgų tinkle vidutiniškai truko 204 ms (4.11 pav.). Pasitaikė šuolių, siekiančių 600 ms ir 1 sekundę, tačiau nepriklausomai nuo atliekamų operacijų, patvirtinimo laikas išliko panašus viso testo metu.

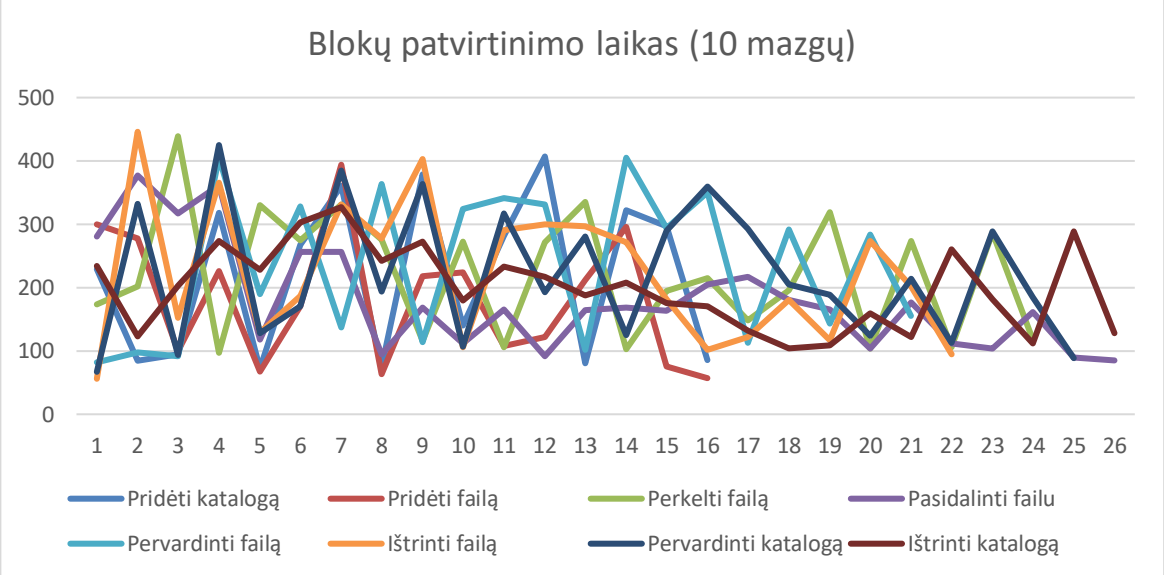
Blokų patvirtinimo laiko dviejų mazgų sistemoje tyrimo duomenys pateikti priede 7.1.8 Blokų patvirtinimo laikas (2 mazgai).



4.11 pav. Blokų patvirtinimo laikas dviejų mazgų sistemoje

Atlikus tyrimą didesnėje sistemoje su dešimt mazgų pastebėta, kad blokų patvirtinimo laikas išliko beveik toks pat, kaip dviejų mazgų sistemoje (4.12 pav.). Vidutiniškai vienas blokas patvirtintas per 209 ms. Šis laikas gali svyruoti ir augti, jei sistemoje dalyvaus labiau nuo vienas kito nutolę mazgai.

Blokų patvirtinimo laiko dešimties mazgų sistemoje tyrimo duomenys pateikti priede 7.1.9 Blokų patvirtinimo laikas (10 mazgų).



4.12 pav. Blokų patvirtinimo laikas dešimties mazgų sistemoje

4.3. Išvados

Tyrimo metu nustatytos realizuoto prototipo charakteristikos. Pateiktos teigiamos ir neigiamos sukurtos failų sistemos savybės. Sistema palyginta su centralizuotos duomenų saugyklos savybėmis.

Kiekybiniam tyrimui sudarytas veiksmų planas. Numatyta tyrimą atlikti skirtingo dydžio sistemose – dviejų ir dešimties mazgų. Kiekviena failų sistemos operacija buvo vykdoma penkiasdešimt kartų. Surinkti bandymų rezultatai ir pamatuotos laiko bei atminties sąnaudos reikalingos sistemos veikimui.

Blokų grandinė nėra laikoma greita technologija, tačiau tyrimo metu pastebėta, kad operacijų transakcijos kuriamos ir tvirtinamos pakankamai greitai. Taip pat ir naujų blokų kūrimas ir jungimas į grandinę vykdomas be didelio uždelsimo. Sistema neatlieka veiksmų akimirksniu, tačiau atsižvelgiant į architektūrą panaudojant blokų grandinių technologiją, galima teigti, kad sistemos našumas geras ir operacijos įvykdomos gana sparčiai.

5. IŠVADOS

Darbo metu pasiūlytas metodas saugiam informacijos saugojimui naudojant blokų grandinių technologiją bei realizuotas ir aprašytas sistemos prototipas. Atlikta analizė, kurioje buvo apžvelgti informacijos saugojimo metodai ir atliktas jų palyginimas. Išanalizuota blokų grandinių technologija, pateikti pranašumai ir trūkumai saugiam informacijos saugojimui.

Pabrėžtos blokų grandinių technologijos teigiamos savybės:

- sutrikdžius vieną mazgą (ar nustojus jam veikti), lieka daug grandinės kopijų visame tinkle, taigi informacija neprarandama ir ją lengva parsiųsti iš naujo;
- sunkus (praktiškai neįmanomas) duomenų klastojimas, nes transakcijos ir blokai validuojami daugelio mazgų pagalba visame tinkle;
- validacija daugelyje mazgų užtikrina sistemos ir duomenų vientisumą;
- į grandinę įrašytų duomenų pakeisti negalima – tai užtikrina duomenų vientisumą;
- tai decentralizuota (paskirstyta) sistema, taigi užtikrinamas pasiekiamumas.

Išsiaiškinta, kad blokų grandinės nėra skirtos didelės apimties duomenims saugoti. Realizuojant saugaus informacijos saugojimo metodą paremtą blokų grandinėmis, reikia turėti omenyje tai, kad norint saugoti didelius failus, privaloma naudoti išorinę saugyklą.

Pagal atliktos analizės išvadas buvo pasiūlytas metodas saugiam informacijos saugojimui naudojant blokų grandinių technologiją. Visų operacijų, atliktų veiksmų, failų metaduomenų istorija turi būti saugoma blokų grandinėje kartu su nuorodomis į konkrečius failus ir taisyklėmis, kaip juos gauti ir grąžinti į pirminę būseną.

Metodo savybės saugiam informacijos saugojimui:

- sistema decentralizuota, nėra vieno tiesos šaltinio, sprendimai priimami autonomiškai tarp visų tinkle veikiančių mazgų;
- paskirstyta sistema užtikrina pasiekiamumą;
- negalima keisti įrašytos informacijos, matoma visa įvykių istorija;
- duomenų patikrinimą atlieka visi sistemos tinklo nariai, tai užtikrinamas vientisumas;
- konfidencialumas užtikrinamas elektroniniu parašu ir pasitelkiant papildomas kriptografines priemones šifruojant transakcijų bei failų turinį.

Realizuotas failų sistemos prototipas, kuris atspindi projekto dalyje pateiktą projekto aprašą ir įgalina apibrėžtas sistemos savybes saugumo aspektu. Pateikta prototipo architektūra ir realizuoti komponentai. Aprašytas įvairių dalių veikimas ir paaiškinti priimti implementacijos sprendimai.

Tyrimo metu nustatytos realizuoto prototipo charakteristikos. Pateiktos teigiamos ir neigiamos sukurtos failų sistemos savybės. Sistema palyginta su centralizuotos duomenų saugyklos savybėmis. Kiekybiniam tyrimui sudarytas veiksmų planas, kurio laikantis buvo atlikti bandymai ir surinkta informacija apie laiko bei atminties sąnaudas sistemoje.

6. LITERATŪRA

[1] P. Hernandez, „Data Storage Security: What Storage Professionals Need to Know,“ www.enterprisestorageforum.com, 19 04 2018. [Tinkle]. Available: https://www.enterprisestorageforum.com/storage-management/data-storage-security-guide.html.

[2] A. Ali ir D. M. Afzal, „Database Security: Threats and Solutions,“ *International Journal of Engineering Inventions,* t. VI, nr. 2, pp. 25-27, 2017.

[3] G. Duncan, „Ditch the shoebox for a vault: How to preserve your digital life for decades,“ Digital Trends, 15 03 2014. [Tinkle]. Available: https://www.digitaltrends.com/computing/keeping-data-safe-eternity/.

[4] L. Alton, „The 7 Biggest Problems in Data Storage – and How to Overcome Them,“ Smart Data Collective, 10 02 2018. [Tinkle]. Available: https://www.smartdatacollective.com/7-biggest-problems-data-storage-overcome/.

[5] R. B. C. M. Julia Palmer, „Magic Quadrant for Distributed File Systems and,“ Gartner, 2019.

[6] R. Rivest, „The MD5 Message-Digest Algorithm,“ MIT Laboratory for Computer Science, 04 1992. [Tinkle]. Available: https://www.ietf.org/rfc/rfc1321.txt.

[7] P. A. DesAutels, „SHA1 Secure Hash Algorithm - Version 1.0,“ 1 10 1997. [Tinkle]. Available: https://www.w3.org/PICS/DSig/SHA1_1_0.html.

[8] A. B. C. D. G. H. K. D. M. Carlo Baliello, „Kerberos protocol: an overview,“ 2002.

[9] A. Balchunas, „Access Control Lists,“ http://www.routeralley.com/, 2007.

[10] J. A. C. D. R. K. David F. Ferraiolo, „Role-Based Access Control (RBAC): Features and Motivations,“ National Institute of Standards and Technology, Gaithersburg.

[11] Z. Zheng, S. Xie, H. Dai, X. Chen ir H. Wang, „An Overview of Blockchain Technology Architecture Consensus and Future Trends,“ įtraukta *2017 IEEE 6th International Congress on Big Data*, 2017.

[12] I. Zikratov, A. Kuzmin, V. Akimenko, V. Niculichev ir L. Yalansky, „Ensuring data integrity using blockchain technology,“ įtraukta *20th Conference of Open Innovations Association (FRUCT)*, Saint-Petersburg, Russia, 2017.

[13] W. W. A. P. David Shrier, „Blockchain & Infrastructure (Identity, Data Security),“ MASSACHUSETTS INSTITUTE OF TECHNOLOGY, Massachusetts, 2016.

[14] G. Greenspan, „Blockchains vs centralized databases,“ MultiChain, 17 03 2016. [Tinkle]. Available: https://www.multichain.com/blog/2016/03/blockchains-vs-centralized-databases/.

[15] S. Khatwani, „Understanding The Difference Between A Database & Blockchain,“ CoinSutra, 15 10 2018. [Tinkle]. Available: https://coinsutra.com/blockchain-vs-database/.

[16] M. B. R. C. H. Krawczyk, „HMAC: Keyed-Hashing for Message Authentication,“ 02 1997. [Tinkle]. Available: https://tools.ietf.org/html/rfc2104.

7. PRIEDAI

7.1. Kiekybinio tyrimo rezultatai

Atminties matavimo vienetas – baitas.
Laiko matavimo vienetas – milisekundės.

7.1.1. Transakcijų dydis

Transakcija	Pridėti katalogą	Pridėti failą	Perkelti failą	Pasidalinti failu	Pervardinti failą	Ištrinti failą	Pervardinti katalogą	Ištrinti katalogą
1	1631	2687	1807	3395	1775	1759	1791	1759
2	1631	2687	1807	3395	1775	1759	1791	1759
3	1631	2687	1807	3395	1775	1759	1791	1759
4	1631	2687	1807	3395	1775	1759	1791	1759
5	1631	2687	1807	3395	1775	1759	1791	1759
6	1631	2687	1807	3395	1775	1759	1791	1759
7	1631	2687	1807	3395	1775	1759	1791	1759
8	1631	2687	1807	3395	1775	1759	1791	1759
9	1631	2687	1807	3395	1775	1759	1791	1759
10	1631	2687	1807	3395	1775	1759	1791	1759
11	1631	2687	1807	3395	1775	1759	1791	1759
12	1631	2687	1807	3395	1775	1759	1791	1759
13	1631	2687	1807	3395	1775	1759	1791	1759
14	1631	2687	1807	3395	1775	1759	1791	1759
15	1631	2687	1807	3395	1775	1759	1791	1759
16	1631	2687	1807	3395	1775	1759	1791	1759
17	1631	2687	1807	3395	1775	1759	1791	1759
18	1631	2687	1807	3395	1775	1759	1791	1759
19	1631	2687	1807	3395	1775	1759	1791	1759
20	1631	2687	1807	3395	1775	1759	1791	1759
21	1631	2687	1807	3395	1775	1759	1791	1759
22	1631	2687	1807	3395	1775	1759	1791	1759
23	1631	2687	1807	3395	1775	1759	1791	1759
24	1631	2687	1807	3395	1775	1759	1791	1759
25	1631	2687	1807	3395	1775	1759	1791	1759
26	1631	2687	1807	3395	1775	1759	1791	1759
27	1631	2687	1807	3395	1775	1759	1791	1759
28	1631	2687	1807	3395	1775	1759	1791	1759
29	1631	2687	1807	3395	1775	1759	1791	1759
30	1631	2687	1807	3395	1775	1759	1791	1759
31	1631	2687	1807	3395	1775	1759	1791	1759
32	1631	2687	1807	3395	1775	1759	1791	1759
33	1631	2687	1807	3395	1775	1759	1791	1759
34	1631	2687	1807	3395	1775	1759	1791	1759
35	1631	2687	1807	3395	1775	1759	1791	1759
36	1631	2687	1807	3395	1775	1759	1791	1759
37	1631	2687	1807	3395	1775	1759	1791	1759

38	1631	2687	1807	3395	1775	1759	1791	1759
39	1631	2687	1807	3395	1775	1759	1791	1759
40	1631	2687	1807	3395	1775	1759	1791	1759
41	1631	2687	1807	3395	1775	1759	1791	1759
42	1631	2687	1807	3395	1775	1759	1791	1759
43	1631	2687	1807	3395	1775	1759	1791	1759
44	1631	2687	1807	3395	1775	1759	1791	1759
45	1631	2687	1807	3395	1775	1759	1791	1759
46	1631	2687	1807	3395	1775	1759	1791	1759
47	1631	2687	1807	3395	1775	1759	1791	1759
48	1631	2687	1807	3395	1775	1759	1791	1759
49	1631	2687	1807	3395	1775	1759	1791	1759
50	1631	2687	1807	3395	1775	1759	1791	1759

7.1.2. Blokų dydis

Blokas	Pridėti katalogą	Pridėti failą	Perkelti failą	Pasidalinti failu	Pervardinti failą	Ištrinti failą	Pervardinti katalogą	Ištrinti katalogą
1	6942	12222	4972	9736	3483	5307	1909	1909
2	3195	2965	2085	3673	7662	4828	7742	3451
3	6942	12222	7822	15762	6269	6205	4924	2037
4	1909	12222	2085	15762	2053	6205	6333	3451
5	6942	12222	4972	6723	6269	3451	2069	2037
6	6942	2965	6397	3673	2053	2037	6333	3451
7	6942	12222	6397	12749	6269	6205	2069	4828
8	1909	2965	2085	3673	2053	3451	4924	3451
9	6942	12222	6397	15762	6269	2037	2069	4828
10	1909	5307	2085	12749	6269	4828	6333	3451
11	5693	12222	4972	3673	6269	2037	4924	3451
12	1909	12222	2085	12749	2053	3451	2069	3451
13	6942	2965	2085	3673	4876	4828	6333	3451
14	6942	9917	2085	12749	2053	3451	4924	2037
15			3547	3673	6269	4828	2069	3451
16			4972	12749	4876	3451	4924	3451
17			3547	9736	6269	2037	2069	3451
18			4972			3451	4924	3451
19			2085			3451	4924	3451
20			4972			2037	3515	3451
21						3451		3451
22						2037		2037
23						2037		2037
24						3451		2037

25						2037		3451
26								3451
27								3451
28								2037
29								2037

7.1.3. Blokų grandinės augimas

Blokas	Pridėti katalogą	Pridėti failą	Perkelti failą	Pasidalinti failu	Pervardinti failą	Ištrinti failą	Pervardinti katalogą	Ištrinti katalogą
1	3095	3095	4972	9736	3483	5307	1909	1909
2	10037	15317	7057	13409	11145	9656	15484	6902
3	13232	21247	14879	29171	17414	17238	17590	7525
4	20174	30504	16964	44933	19467	23443	25332	12390
5	22083	42726	21936	51656	25736	24140	23137	13013
6	29025	54948	28333	55329	27789	24763	33734	17878
7	35967	57913	34730	68078	34058	35136	31539	24083
8	42909	70135	36815	71751	36111	35833	39318	26157
9	44818	73100	43212	87513	42380	36456	38532	32362
10	51760	85322	45297	100262	48649	44075	49129	34436
11	55578	90629	50269	103935	54918	43321	52644	37887
12	59362	102851	52354	116684	56971	48186	51858	41338
13	61271	115073	54439	120357	61847	54391	62455	44789
14	68213	118038	56524	133106	63900	56465	65970	45412
15	75155	127955	60071	136779	70169	62670	65184	50277
16			65043	149528	75045	64744	72963	53728
17			68590	162937	81314	65367	72177	57179
18			73562			70232	79956	60630
19			75647			73683	84880	64081
20			85591			74306	86986	67532
21						79171		70983
22						79794		71606
23						81831		73643
24						86696		75680
25						87319		80545
26								83996
27								87447
28								88070
29								90107

7.1.4. Transakcijų sukūrimo laikas

Transakcija	Pridėti katalogą	Pridėti failą	Perkelti failą	Pasidalinti failu	Pervardinti failą	Ištrinti failą	Pervardinti katalogą	Ištrinti katalogą
1	17	22	13	17	13	13	15	14
2	16	14	13	13	12	13	13	12
3	16	15	13	13	13	12	13	12
4	17	14	15	13	12	13	13	12
5	15	13	13	13	12	15	13	12

6	14	16	13	13	12	12	13	13
7	15	13	17	13	12	16	13	12
8	14	15	16	12	12	13	14	12
9	13	20	16	14	15	13	13	12
10	13	16	13	13	17	14	14	12
11	14	14	13	14	11	12	14	13
12	16	13	12	13	15	16	13	11
13	14	17	13	12	12	13	13	12
14	13	14	15	13	11	13	13	16
15	14	13	13	14	11	12	13	11
16	17	15	13	13	14	14	13	12
17	13	17	14	15	14	13	13	11
18	13	13	14	15	12	12	15	12
19	16	13	13	15	13	13	17	12
20	14	13	12	13	15	12	14	12
21	12	12	13	14	13	12	13	12
22	13	13	14	14	13	13	14	12
23	13	14	13	13	13	12	13	15
24	13	13	13	14	13	13	13	11
25	25	14	16	12	12	12	13	13
26	15	13	13	14	12	15	13	12
27	12	15	13	13	12	12	13	13
28	12	13	15	13	12	12	14	13
29	13	13	13	13	12	14	15	12
30	15	13	14	12	12	13	13	16
31	19	13	12	13	12	13	13	12
32	13	13	16	16	13	13	15	13
33	13	13	13	12	11	14	15	12
34	15	13	13	12	12	13	12	12
35	12	14	13	13	20	13	13	12
36	13	14	12	13	12	13	12	16
37	15	14	14	12	13	12	13	12
38	13	12	15	13	11	12	12	18
39	12	12	13	13	15	15	12	13
40	12	13	15	12	12	13	12	12
41	13	13	15	13	12	13	14	13
42	13	13	12	12	12	13	16	12
43	13	13	14	14	13	15	20	15
44	14	12	14	13	12	13	13	12
45	13	12	14	13	12	13	13	13
46	15	13	16	13	11	13	15	12
47	12	12	13	13	14	13	14	12
48	14	13	13	12	12	13	16	11
49	12	13	13	15	12	14	14	13
50	12	14	15	13	12	16	17	13

7.1.5. Transakcijų patvirtinimo laikas (2 mazgai)

Transakcija	Pridėti katalogą	Pridėti failą	Perkelti failą	Pasidalinti failu	Pervardinti failą	Ištrinti failą	Pervardinti katalogą	Ištrinti katalogą
1	24	24	16	19	16	15	18	17
2	20	18	16	17	16	15	16	15
3	20	18	17	17	16	15	16	14
4	20	18	19	16	15	16	15	15
5	19	17	15	16	15	21	16	15
6	18	22	16	21	20	15	16	15
7	18	17	23	16	16	21	17	14
8	18	18	22	15	15	16	17	14
9	17	24	21	18	20	16	15	14
10	17	20	19	16	22	17	18	14
11	18	16	16	16	15	15	20	15
12	20	16	15	18	24	20	16	14
13	17	21	16	15	16	15	18	14
14	16	15	18	16	14	16	16	22
15	17	16	17	18	14	15	17	14
16	25	22	16	17	16	17	16	15
17	16	20	17	18	17	15	16	14
18	16	16	16	18	15	16	17	14
19	20	15	20	21	15	16	19	15
20	17	16	14	18	18	15	16	14
21	15	15	14	20	16	15	16	15
22	15	16	18	17	16	15	17	14
23	16	17	16	16	16	15	16	18
24	16	16	19	17	15	16	17	14
25	36	16	18	16	15	15	16	16
26	18	15	16	20	15	19	16	15
27	15	18	16	16	14	15	18	15
28	15	15	17	15	15	16	17	15
29	17	16	16	16	14	17	18	15
30	19	17	16	16	15	16	15	19
31	24	15	15	15	15	15	16	14
32	16	16	19	19	16	17	18	16
33	15	15	16	15	14	17	19	14
34	18	15	15	16	14	15	15	15
35	15	16	16	16	23	16	15	15
36	16	17	16	16	15	15	15	20
37	17	16	16	17	15	16	16	15
38	16	15	19	16	14	15	15	21
39	15	15	16	15	18	19	14	14
40	16	16	17	16	15	16	15	14
41	16	15	18	16	15	15	17	15

42	16	15	16	15	16	16	18	14
43	16	15	16	17	15	19	24	20
44	16	15	17	16	14	16	17	15
45	15	15	18	16	15	17	16	15
46	18	15	18	16	14	16	18	15
47	15	15	16	17	17	16	18	15
48	17	16	16	15	15	15	19	14
49	15	17	16	17	14	16	17	17
50	15	17	16	16	15	19	20	19

7.1.6. Transakcijų patvirtinimo laikas (10 mazgų)

Transakcija	Pridėti katalogą	Pridėti failą	Perkelti failą	Pasidalinti failu	Pervardinti failą	Ištrinti failą	Pervardinti katalogą	Ištrinti katalogą
1	2309	2396	24	46	23	21	20	24
2	1290	1417	30	32	28	19	22	22
3	252	399	25	27	21	29	20	25
4	38	28	22	24	24	21	22	45
5	35	29	24	25	21	21	22	23
6	28	28	27	24	23	24	21	21
7	24	37	21	39	41	22	20	30
8	50	25	23	25	23	21	19	28
9	23	24	22	22	21	22	24	23
10	29	28	25	22	22	23	22	22
11	27	28	33	22	25	22	22	26
12	24	27	21	24	23	22	20	23
13	23	33	34	24	24	23	21	21
14	22	23	24	31	20	24	21	20
15	21	23	22	22	23	21	20	21
16	22	22	28	22	23	20	22	26
17	48	42	23	22	26	20	21	22
18	22	24	21	25	26	19	21	23
19	22	25	30	23	20	22	21	21
20	22	22	28	22	24	24	20	21
21	22	25	22	26	22	33	22	28
22	24	23	22	31	23	23	21	22
23	25	31	21	24	24	20	22	20
24	22	41	32	23	45	20	21	25
25	21	52	21	23	21	22	21	21
26	20	23	24	37	30	32	21	20
27	21	23	22	46	23	24	25	23
28	23	24	21	21	25	25	21	28
29	22	24	21	22	22	21	21	24
30	22	75	20	51	22	21	19	22
31	23	25	29	19	23	22	20	28
32	23	27	20	21	22	21	19	24

33	22	30	22	22	22	23	20	22
34	72	25	20	21	32	21	23	19
35	23	24	21	22	25	23	20	23
36	20	24	23	23	26	44	19	21
37	22	33	20	22	25	20	27	20
38	67	23	21	22	21	38	21	19
39	20	19	23	21	22	23	23	22
40	22	21	25	21	19	46	22	20
41	22	23	21	20	21	33	28	25
42	24	28	25	23	25	43	34	28
43	22	28	23	22	25	24	20	27
44	23	49	20	21	37	25	20	23
45	32	33	21	23	23	22	29	21
46	22	27	22	22	22	20	46	19
47	25	27	22	21	21	21	23	20
48	25	27	24	23	21	21	24	23
49	22	24	21	28	25	21	46	19
50	26	24	24	21	22	23	20	46

7.1.7. Blokų kasimos laikas

Blokas	Pridėti katalogą	Pridėti failą	Perkelti failą	Pasidalinti failu	Pervardinti failą	Ištrinti failą	Pervardinti katalogą	Ištrinti katalogą
1	807	235	1497	440	592	71	329	392
2	131	696	415	10	244	316	36	365
3	32	48	288	88	206	100	32	273
4	93	758	77	842	89	813	1676	46
5	362	199	97	141	135	248	373	164
6	44	130	122	498	488	472	42	428
7	610	264	81	999	223	361	227	1495
8	105	403	386	420	395	49	173	215
9	772	79	522	584	365	1618	82	617
10	191	1856	1165	533	151	594	265	165
11	317	1114	75	1025	179	762	1830	727
12	592	120	59	221	658	518	947	6
13	389	306	84	52	681	1230	1342	376
14	179	480	614	287	131	297	61	250
15	238	108	153	308	28	683	178	46
16	305	435	284	781	1174	251	837	158
17			206	626	364	633	329	59
18			141	256	342	45	44	350

19			339	308	166	147	982	404
20			177	45	666	300	203	37
21			185	70	204	997	1430	10
22			107	98		555	1023	676
23			2510	77			86	422
24			9	1			169	248
25				1			92	573
26				439				43
27								103
28								37
29								85
30								159

7.1.8. Blokų patvirtinimo laikas (2 mazgai)

Blokas	Pridėti katalogą	Pridėti failą	Perkelti failą	Pasidalinti failu	Pervardinti failą	Ištrinti failą	Pervardinti katalogą	Ištrinti katalogą
1	496	165	308	240	157	73	66	63
2	204	46	92	81	376	262	410	185
3	1045	154	497	373	312	346	242	91
4	44	158	105	372	81	327	346	156
5	120	157	324	152	286	200	87	98
6	14	39	375	75	79	87	384	171
7	171	155	380	287	285	335	86	260
8	392	37	101	79	77	163	259	158
9	673	165	370	365	291	102	81	255
10	72	69	98	291	299	247	315	179
11	119	168	280	75	306	90	250	178
12	669	184	98	291	82	176	82	204
13	103	41	105	77	237	269	332	156
14	583	167	96	287	81	178	279	90
15			199	77	302	269	86	162
16			288	291	231	166	283	185
17			187	241	289	84	87	177
18			322			184	258	182
19			119			162	259	177
20			289			85	227	190
21						181		187

22						83		91
23						87		92
24						224		123
25						92		197
26								198
27								159
28								107
29								103

7.1.9. Blokų patvirtinimo laikas (10 mazgų)

Blokas	Pridėti katalogą	Pridėti failą	Perkelti failą	Pasidalinti failu	Pervardinti failą	Ištrinti failą	Pervardinti katalogą	Ištrinti katalogą
1	229	300	173	281	82	56	67	234
2	85	278	202	377	98	446	332	123
3	95	100	439	317	92	152	94	203
4	318	226	97	362	405	366	425	274
5	72	67	330	118	190	129	128	228
6	268	170	275	256	328	187	170	303
7	360	394	331	256	137	331	385	327
8	74	63	276	95	364	278	194	242
9	379	218	116	168	114	403	364	273
10	140	224	273	112	324	106	107	180
11	282	108	106	165	341	291	317	233
12	407	122	272	92	331	300	193	217
13	80	212	335	164	102	297	281	188
14	322	296	103	168	405	272	125	208
15	297	75	195	163	294	182	290	176
16	86	57	215	205	350	102	359	170
17			148	217	113	122	293	132
18			196	181	292	181	205	104
19			319	165	143	117	189	109
20			104	104	284	274	124	159
21			274	177	155	202	214	122
22			106	112		95	113	260
23			287	104			289	182
24			117	161			185	112
25				90			89	289
26				85				128
27								270
28								160
29								235
30								106