



UNIVERZITET SINGIDUNUM

Departman za poslediplomske studije

DOKTORSKE STUDIJE

Napredni sistemi zaštite

Unapređenje distribuirane platforme
za e-trgovinu na javnoj *blockchain*
mreži u cilju minimizacije
računarskog napora prilikom
izvršavanja transakcija

- doktorska disertacija -

Mentor:

prof. dr Mladen Veinović

Student:

Vukašin S. Tasić 2020/460030

Beograd, 2026. godina

Zahvalnica

Zahvaljujem se svom mentoru, prof. dr Mladenu Veinoviću, na ukazanom poverenju, stručnoj podršci, korisnim savetima i kontinuiranom usmeravanju tokom studija i izrade ove doktorske disertacije.

Posebnu zahvalnost upućujem svom prijatelju Branislavu Đaliću. Kroz dugogodišnje prijateljstvo i zajednički rad na velikom broju projekata imao sam priliku da naučim mnogo o blokčejn tehnologijama, njihovoj praktičnoj primeni i izazovima industrije. Znanje i iskustvo koje je nesebično delio sa mnom imali su značajan uticaj na moje profesionalno usmerenje i razvoj.

Najveću zahvalnost dugujem svojoj porodici, ocu Srđanu, majci Ireni i bratu Nemanji, na razumevanju, strpljenju i безусловnoj podršci tokom svih godina studiranja. Njihova podrška predstavljala je oslonac u trenucima izazova i bila neophodna za uspešan završetak ovog rada.

Sažetak

Elektronska trgovina predstavlja jedan od najznačajnijih domena primene informacionih sistema. Savremene platforme omogućavaju visok nivo funkcionalnosti i upotrebljivosti, ali istovremeno uvode ograničenja u pogledu transparentnosti, autonomije korisnika i raspodele kontrole nad sistemom, budući da se zasnivaju na centralizovanom modelu poverenja.

Predmet istraživanja ove disertacije jeste analiza mogućnosti primene javnih blokčejn sistema kao infrastrukturne osnove za realizaciju poslovne logike platformi za elektronsku trgovinu, kao i ispitivanje mogućnosti primene sadržajno adresiranih mreža za decentralizovano skladištenje podataka u cilju povećanja autonomije korisnika.

Polazeći od teorijske analize savremenih informacionih tehnologija, koncepta decentralizacije i blokčejn sistema, razvijen je referentni model za evaluaciju mogućnosti primene javnih blokčejn mreža u elektronskoj trgovini.

Međutim, izvršavanje operacija na ovim sistemima podrazumeva transakcione troškove nastale usled korišćenja računarskih resursa mreže, što predstavlja značajno ograničenje njihove šire primene u elektronskoj trgovini.

S obzirom na dominantnu ulogu *Ethereum* mreže u razvoju decentralizovanih aplikacija, istraživanje je usmereno na dizajn platforme zasnovane na ovoj tehnologiji kroz analizu i optimizaciju modela reprezentacije entiteta elektronske trgovine, sa posebnim fokusom na smanjenje troškova izvršavanja osnovnih operacija nad entitetima sa jedinstvenim identifikatorima.

Na osnovu modifikacije postojećih standarda za nezamenljive tokene predložen je model optimizovanog pakovanja podataka unutar jedne skladišne jedinice, sa ciljem smanjenja broja pristupa trajnom skladištu kao dominantnog izvora troškova izvršavanja.

Eksperimentalna evaluacija sprovedena u kontrolisanom okruženju pokazuje značajne razlike u troškovima operacija kreiranja, prenosa i uništavanja tokena između posmatranih standarda. Rezultati potvrđuju da optimizacija strukture podataka i racionalno korišćenje skladišnih resursa mogu značajno unaprediti ekonomsku efikasnost decentralizovanih aplikacija.

Na osnovu teorijske analize, konstruktivnog modelovanja i eksperimentalnih rezultata predložen je arhitektonski okvir za razvoj decentralizovanih platformi elektronske trgovine koji integriše *Ethereum* kao sloj za realizaciju poslovne logike i *IPFS* kao sloj za skladištenje i distribuciju sadržaja.

Ključne reči: elektronska trgovina, blokčejn, Eterijum, pametni ugovori, NFT, optimizacija skladištenja podataka, IPFS, decentralizovane aplikacije.

Abstract

Electronic commerce represents one of the most significant application domains of information systems. Modern platforms provide a high level of functionality and usability, but they simultaneously introduce limitations regarding transparency, user autonomy, and the distribution of control within the system, as they are predominantly based on a centralized trust model.

The subject of this dissertation is the analysis of the applicability of public blockchain systems as an infrastructural foundation for implementing the business logic of electronic commerce platforms, as well as the examination of content-addressed networks for decentralized data storage with the aim of increasing user autonomy.

Starting from a theoretical analysis of modern information technologies, the concept of decentralization, and blockchain systems, a reference model for evaluating the applicability of public

blockchain networks in electronic commerce was developed.

However, the execution of operations on such systems involves transaction costs arising from the consumption of network computational resources, which represents a significant limitation to their wider adoption in electronic commerce.

Considering the dominant role of *Ethereum* in the development of decentralized applications, the research is focused on the design of a platform based on this technology through the analysis and optimization of electronic commerce entity representation models, with particular emphasis on reducing the execution costs of fundamental operations on uniquely identifiable entities.

Based on modifications of existing non-fungible token standards, an optimized data-packing model within a single storage slot is proposed, aiming to reduce the number of accesses to persistent storage, which constitutes the dominant source of execution costs.

Experimental evaluation conducted in a controlled environment demonstrates significant differences in the costs of token minting, transfer, and burning operations among the observed standards. The results confirm that data structure optimization and efficient utilization of storage resources can significantly improve the economic efficiency of decentralized applications.

Based on theoretical analysis, constructive modeling, and experimental results, an architectural framework for the development of decentralized electronic commerce platforms is proposed, which integrates *Ethereum* as the business logic execution layer and *IPFS* as the content storage and distribution layer.

Keywords: electronic commerce, blockchain, Ethereum, smart contracts, NFT, data storage optimization, IPFS, decentralized applications.

Sadržaj

1	Metodološki okvir disertacije	1
1.1	Obrazloženje predložene teze	1
1.2	Predmet istraživanja	2
1.3	Problem istraživanja	3
1.4	Ciljevi istraživanja	4
1.5	Hipoteze istraživanja	5
1.6	Metodološki pristup	7
1.7	Eksperimentalni metod	8
1.8	Ograničenja predložene metodologije	10
1.9	Naučni doprinos	11
1.10	Struktura disertacije	11
2	Savremeni informacioni sistemi	14
2.1	Radni sistem	14
2.2	Informacioni sistem	16
2.2.1	Tjuringova kompletnost	17
2.3	Informacioni sistem elektronske trgovine	18
2.4	Arhitektonska organizacija informacionih sistema	19
2.4.1	Računarstvo u oblaku i elastično skaliranje informacionih sistema	23
2.5	Centralizacija u distribuiranim informacionim sistemima	24
2.5.1	Arhitektonska decentralizacija	25
2.5.2	Kontrolna decentralizacija	26
2.5.3	Logička decentralizacija	27
2.6	Granice i implikacije decentralizacije u distribuiranim informacionim sistemima	28
3	Analitički okvir za evaluaciju informacionih sistema	29
3.1	CAP teorema	29
3.2	Performanse i efikasnost	30
3.2.1	Vertikalno skaliranje	32
3.2.2	Lokacijski zasnovano adresiranje	33
3.2.3	Horizontalno skaliranje	33
3.2.4	Teorija distribuiranih sistema	35
3.2.5	Konsenzus	37
3.2.6	Analiza troškova sinhronizacije i kontencije	40
3.3	Operativna svojstva	41

3.3.1	Kriptografski i poverljivi mehanizmi	42
3.3.1.1	Simetrična kriptografija	42
3.3.1.2	Asimetrična kriptografija	43
3.3.1.3	Hibridna kriptografija	45
3.3.1.4	Kriptografske heš funkcije	47
3.3.2	Funkcionalni bezbednosni okvir	47
3.3.2.1	Autentikacija	48
3.3.2.2	Autorizacija	49
3.3.2.3	Evidencija	50
3.3.3	Bezbednosne pretnje i napadi	51
4	Blokčejn kao infrastrukturni sloj informacionih sistema	52
4.1	Geneza blokčejn tehnologije	53
4.1.1	Transakcije i blokovi	56
4.1.1.1	Performanse blokčejn sistema	58
4.1.1.2	Ograničenja protokola u Bitcoin mreži	59
4.1.2	Sistem postizanja konsenzusa i validacija novih blokova	59
4.1.2.1	Algoritam dokazivanja uložnog rada	61
4.1.2.2	Verovatnosni konsenzus i praktična finalnost transakcija	62
4.1.3	Struktura i funkcionalnost blokčejn sistema	63
4.1.3.1	Bitcoin Script kao mehanizam kontrole pristupa	64
4.1.3.2	Pametni ugovori	66
4.1.3.3	Apstrakcija naloga	67
4.2	Upravljanje blokčejnom	68
4.2.1	Decentralizovane autonomne organizacije	69
4.3	Blokčejn trilema	69
4.3.1	Decentralizovanost blokčejn sistema	70
4.3.2	Sigurnost blokčejn sistema	72
4.3.3	Skalabilnost blokčejn sistema	73
5	Arhitektura Eterijum blokčejna	74
5.1	Konsenzusni sloj Eterijum mreže	76
5.1.1	Račvanja u konsenzusnom sloju	78
5.1.2	Mehanizam finalizacije blokova	79
5.1.3	Bezbednosni mehanizmi konsenzusa	80
5.1.3.1	Mehanizam sečenja zlonamernih validatora	81
5.1.3.2	Napadi sa malom težinom uloga	81
5.1.3.3	Napadi sa velikom težinom uloga	82
5.1.3.4	Društveni sloj Eterijuma	83
5.2	Izvršni sloj Eterijum mreže	84
5.2.1	Skladište naloga	84
5.2.2	Virtuelna memorija samo za čitanje	87
5.2.3	Stanje mašine	88
5.3	Transakcije i njihova obrada u Eterijumu	90

6	Metodologija optimizacije pametnih ugovora za e-trgovinu	93
6.1	Troškovi pristupa skladištu i njihov uticaj na cenu izvršavanja	93
6.2	Modelovanje vlasništva nad digitalnim entitetima na blokčejnu u sistemima e-trgovine	95
6.3	Enkodirana reprezentacija entiteta sa jedinstvenim identifikatorom u okviru jedne skladišne jedinice	96
6.3.1	ERC-721	98
6.3.1.1	Kreiranje	99
6.3.1.2	Prenos	100
6.3.1.3	Uništavanje	100
6.3.2	ERC-721A	101
6.3.2.1	Kreiranje	102
6.3.2.2	Prenos	103
6.3.2.3	Uništavanje	106
6.3.3	ERC-721Psi	106
6.3.3.1	Kreiranje	107
6.3.3.2	Prenos	109
6.3.3.3	Uništavanje	111
6.4	Rezultati eksperimentalne analize	112
7	Arhitektura IPFS mreže	118
7.1	Strukturiranje i adresiranje sadržaja	119
7.2	Distribuirano pribavljanje sadržaja	120
7.3	Dugoročno skladištenje podataka	122
7.4	Korisnički pristup decentralizovanim aplikacijama	123
8	Diskusija	125
8.1	Proširene funkcionalnosti NFT standarda u sistemima e-trgovine	125
8.2	Obrasci modularnog dizajna pametnih ugovora	127
8.3	Integracija blokčejn sistema sa spoljašnjim izvorima podataka	128
8.4	Decentralizovano skladištenje podataka	129
9	Zaključak	131

Glava 1

Metodološki okvir disertacije

1.1 Obrazloženje predložene teze

Elektronska trgovina definiše se kao proces prodaje ili kupovine robe i usluga realizovan putem računarskih mreža korišćenjem metoda posebno projektovanih za prijem ili izdavanje porudžbina. U osnovi svakog sistema elektronske trgovine nalazi se poslovni model koji definiše proizvode i usluge predstavljene jedinstvenim identifikatorima, model ponašanja kupca koji se bavi komunikacijom između korisnika i platforme, i infrastruktura neophodna za realizaciju i podršku sistemu elektronske trgovine. Korisnici predstavljaju osnovni izvor prihoda sistema elektronske trgovine i korisničko iskustvo direktno zavisi od performansi informacione infrastrukture na kojoj se sistem zasniva.

Savremeni informacioni sistemi, uključujući sisteme elektronske trgovine, dominantno su zasnovani na centralizovanom modelu u kojem se poverenje delegira trećoj strani. Ovakav pristup, iako funkcionalan, uvodi određena ograničenja u obliku asimetrije moći između platformi i njihovih korisnika. Centralizacija podrazumeva da treća strana upravlja podacima korisnika i posreduje u obradi plaćanja, čime preuzima ključnu ulogu u sistemu i održava privatnu knjigu evidencije. Iako ovakav pristup obezbeđuje pozitivno korisničko iskustvo, naročito u domenu plaćanja, troškovi posredovanja nisu uvek transparentni krajnjim korisnicima. Pored toga, oslanjanje na centralizovane entitete može dovesti do problema kao što su zloupotreba podataka, netransparentnost i sistemski rizici.

Pojava *Bitcoin* (Bitcoin) tehnologije predstavlja značajan iskorak u ovom domenu, jer uvodi model elektronskog plaćanja zasnovan na kriptografskom dokazu umesto poverenja. Na taj način omogućava se direktno izvršavanje transakcija između učesnika, bez potrebe za centralnim posrednicima. Sistem se zasniva na javnoj dostupnosti zapisa o svim transakcijama izvršenim na mreži i njihovom redosledu kroz javnu distribuiranu knjigu evidencije koju održava decentralizovana mreža čvorova, čime se obezbeđuju transparentnost i verifikabilnost.

Eliminacijom centralnog posrednika javlja se problem koordinacije i postizanja konsenzusa o budućem stanju sistema u otvorenom okruženju jer može uključivati zlonamerne aktore. Ovaj problem rešava se Nakamoto konsenzusnim mehanizmom, koji uvodi oskudan resurs potreban za predlaganje novog stanja mreže. Time se uspostavlja asimetrija računarske složenosti, jer pronalaženje rešenja jedino je moguće značajnim naporom zasnovanim na metodama grube sile, dok je verifikacija pronađenog rešenja brza i deterministička, čime se obezbeđuje sigurnost sistema.

Prilikom kreiranja transakcije korisnik dokazuje pravo raspolaganja sredstvima u skladu sa uslovima definisanim u samoj transakciji, pri čemu se ulazi transakcije u potpunosti troše, čime se rešava problem dvostruke potrošnje. Ovo se ostvaruje putem ugrađenog skriptnog jezika, koji predstavlja

sastavni deo transakcione strukture i omogućava definisanje jednostavnih, determinističkih uslova izvršavanja bez potrebe za pouzdanim posrednicima. Ovakav pristup konceptualno je blizak ideji samostalno izvršavajućih ugovornih uslova, koju je još 1994. godine formalno definisao Nick Szabo.

Dalji razvoj blokčejn tehnologija doveo je do pojave platforme *Ethereum* (Eterijum), koja integriše koncept distribuirane knjige i pametnih ugovora, uvodeći Turing-kompletnu globalnu distribuiranu računarsku platformu. Pametni ugovori predstavljaju programe na blokčejnu u obliku niza bajtnih instrukcija raspoređenih na određenoj adresi, čijim se izvršavanjem omogućava razvoj decentralizovanih aplikacija, koje omogućavaju automatizaciju poslovne logike i direktnu interakciju među korisnicima.

Primena decentralizovanih aplikacija na javnim blokčejn mrežama zadovoljava poslovni model elektronske trgovine i predstavlja značajan potencijal u ovom domenu. Ovakvi sistemi obezbeđuju visok nivo transparentnosti, integriteta, dostupnosti, autentičnosti i neporecivosti. Dodatno, omogućena su direktna plaćanja između učesnika sistema bez potrebe za posrednicima, čime se smanjuju troškovi posredovanja i pojednostavljuje realizacija poslovnih procesa. Pametni ugovori predstavljaju autonomne programe koji se izvršavaju u distribuiranom okruženju i omogućavaju formalizaciju i automatizaciju pravila sistema.

1.2 Predmet istraživanja

Predmet ovog istraživanja obuhvata analizu blokčejn sistema kao specifične klase distribuiranih sistema, sa ciljem utvrđivanja da li oni mogu da zadovolje operativna svojstva savremenih informacionih sistema. Poseban fokus stavlja se na ispitivanje uticaja decentralizacije, kao proširenja sociotehničke perspektive informacionih sistema, na složenost sistema i potrebu za novim mehanizmima koordinacije koji omogućavaju pouzdano upravljanje zajedničkim stanjem bez centralnog autoriteta.

U radu se razmatra geneza blokčejn tehnologije, kao i klasifikacija blokčejn sistema prema različitim kriterijumima, uključujući model replikacije, otvorenost sistema, kontrolu pristupa, konsenzusni mehanizam, model upravljanja stanjem, stepen sinhronizacije izvršavanja i model upravljanja sistemom, što ima značajan uticaj na dizajn platforme. Posebna pažnja posvećena je analizi osnovnih svojstava, kao što su decentralizacija, sigurnost i performanse, koja predstavljaju ključne kriterijume i ograničenja za evaluaciju javnih blokčejn sistema.

Eterijum blokčejn koristi se kao reprezentativan slučaj za analizu i implementaciju predloženog rešenja poslovne logike platforme za e-trgovinu, budući da predstavlja jedan od najrazvijenijih blokčejn sistema za razvoj decentralizovanih aplikacija. S obzirom na to da ova mreža standardizuje reprezentovanje entiteta putem identifikatora i definisanje osnovnih operacija nad njima u okviru jedinstvenog pametnog ugovora, ovakav pristup je posebno relevantan za sisteme elektronske trgovine, gde svaki proizvod ili usluga mora posedovati jedinstvenu identifikaciju.

Jedan od centralnih aspekata eksperimentalnog istraživanja odnosi se na troškovni model javnih blokčejn sistema, koji uvode eksplicitne transakcione naknade. U tom kontekstu analiziraju se predvidivi troškovi povezani sa reprezentacijom entiteta i izvršavanjem osnovnih operacija nad njima, kao i nepredvidivi troškovi koji proizlaze iz dodatnih funkcionalnosti i složenosti definisanih poslovnim zahtevima.

Fokus istraživanja usmeren je na entitete sa jedinstvenim identifikatorima, koji predstavljaju najopštiji model objekata sa identitetom, vlasništvom i atributima, pri čemu se istražuju mogućnosti optimizacije skladištenja mape vlasništva u postojećim standardima za nezamenljive tokene, radi

efikasnije reprezentacije podataka.

Analiza obuhvata evaluaciju različitih pristupa enkodiranju atributa, gde kuponi predstavljaju reprezentativan studijski slučaj, kao i ograničenja u pogledu gubitka informacija kod takvih enkodiranja. Posebna pažnja posvećena je analizi osnovnih operacija nad entitetima, kao što su kreiranje, prenos i uništavanje, uz poređenje troškova izvršavanja u različitim scenarijima, sa ciljem unapređenja efikasnosti sistema.

Poslovni zahtevi sistema elektronske trgovine mogu se formalizovati kroz različite funkcionalne zahteve, čija se troškovna procena ne može unapred u potpunosti odrediti zbog njihove složenosti i brojnosti. Imajući u vidu da je Eterijum virtuelna mašina Tjuring-kompletna, u radu se analiziraju proširenja pametnih ugovora za podršku funkcionalnostima elektronske trgovine, kao i primena modularnog dizajna koji omogućava nadogradnju postojeće platforme i proširenje funkcionalnosti.

Na kraju, razmatra se primena IPFS mreže kao sloja za sadržajno adresirano i dugoročno skladištenje podataka, kao i decentralizovanog prezentacionog sloja koji posreduje u komunikaciji između korisnika i pametnih ugovora platforme, uz očuvanje suvereniteta korisnika nad kriptografskim identitetom, čime se upotpunjuje arhitektura decentralizovane aplikacije.

1.3 Problem istraživanja

Uprkos navedenim prednostima, primena javnih blokčejn mreža u elektronskoj trgovini suočava se sa značajnim ograničenjima. Ključni problem ogleda se u činjenici da troškovi sistema, koji su u tradicionalnim modelima implicitno uključeni kroz korišćenje posredničkih servisa, u ovom slučaju postaju eksplicitni kroz transakcione naknade koje snose korisnici prilikom izvršavanja operacija na blokčejnu, što uvodi dodatnu složenost iz perspektive korisnika i negativno utiče na ukupni kvalitet korisničkog iskustva.

Kako se blokčejn posmatra kao sistem zasnovan na odnosu ponude i potražnje, gde ponudu čini raspoloživa količina računarskih resursa mreže, dok potražnja odgovara ukupnom zahtevu za njihovim korišćenjem, troškovi izvršavanja transakcija zavise od nivoa opterećenja mreže. To predstavlja značajno ograničenje za primenu blokčejn tehnologija u elektronskoj trgovini, budući da korisnici nisu navikli na eksplicitno plaćanje naknada za korišćenje usluga. Dodatno, neoptimizovan bajtkod pametnih ugovora dovodi do povećane potrošnje gasa, usled čega korisnici plaćaju veće naknade za izvršavanje funkcionalno identičnih operacija, naročito u uslovima visoke iskorišćenosti mreže.

Posebno značajan faktor u ovom kontekstu predstavlja trajno skladištenje podataka, budući da svaki podatak upisan u skladište mora biti lokalno sačuvan od strane svih čvorova mreže, čime se povećava ukupna količina podataka koju sistem mora da održava. Dodatno, operacije pristupa skladištu zahtevaju ulazno-izlazne operacije, što uvodi dodatno režijsko opterećenje i čini operacije čitanja i pisanja među najskupljim operacijama u pogledu potrošnje gasa.

U kontekstu reprezentacije proizvoda i usluga elektronske trgovine korišćenjem standarda za nezamenljive tokene, moguće je u mapi vlasništva iskoristiti neiskorišćen prostor za enkodiranje dodatnih atributa. Međutim, ovaj pristup je ograničen karakteristikama virtuelne mašine, pre svega veličinom skladišnih jedinica. Ograničen broj raspoloživih bitova nameće potrebu za kompromisom između količine i preciznosti informacija koje se enkodiraju, što može dovesti do gubitka informacija ili nemogućnosti reprezentacije složenijih struktura podataka.

Dodatno, primena standarda za nezamenljive tokene koji koriste model implicitnog vlasništva uvodi specifične troškovne karakteristike. U ovim modelima, vlasništvo se ne čuva eksplicitno za svaki pojedinačni entitet, već se razrešava tokom operacija prenosa i uništavanja tokena, što dovodi

do dinamičke prirode troškova izvršavanja u zavisnosti od ulaznih parametara. Dodatno, model zasnovan na implicitnom vlasništvu nameće ograničenje da entiteti kreirani u okviru iste sekvence dele zajedničke attribute, što smanjuje fleksibilnost u modelovanju proizvoljnih objekata.

Prilikom proširenja funkcionalnosti pametnih ugovora za e-trgovinu kroz modularni dizajn uvode se novi izazovi, pri čemu svaka dodatna funkcionalnost povećava složenost sistema i troškove postavljanja pametnih ugovora za kompanije i preduzeća. Takođe, uvođenje novih funkcionalnosti često zahteva dodatne provere i pristup skladištu, kao i povećanje broja izvršnih instrukcija, što direktno utiče na povećanje troškova u odnosu na model iz eksperimentalnog dela i otežava preciznu procenu ukupnih troškova unapred.

S obzirom na to da blokčejn sistemi funkcionišu kao zatvorena i deterministička izvršna okruženja, neophodno je uvesti dodatne mehanizme za integraciju sa spoljašnjim izvorima podataka. Ove informacije mogu biti javne ili privatne i često su neophodne za izvršavanje složenijih poslovnih procesa, uključujući validaciju eksternih događaja ili generisanje nasumičnih vrednosti. Međutim, oslanjanje na spoljašnje izvore uvodi dodatne bezbednosne rizike, jer kompromitovan ili nepouzdan izvor može negativno uticati na izvršavanje pametnih ugovora.

Jedan od fundamentalnih izazova blokčejn sistema predstavlja kompromis između decentralizacije, sigurnosti i skalabilnosti, poznat kao blokčejn trilema, koji neminovno utiče na platformu za e-trgovinu čija se poslovna logika nalazi na blokčejnu. Javne blokčejn mreže daju prioritet decentralizaciji i sigurnosti, što dovodi do ograničenja u pogledu skalabilnosti. Ovaj problem postaje posebno izražen u scenarijima masovne primene, zbog čega je neophodno analizirati postojeće pristupe njegovom rešavanju i pravce daljeg razvoja.

Na kraju, dodatni izazov predstavlja dinamičan razvoj blokčejn tehnologija, koji otežava dugoročno predviđanje pravaca razvoja. Ipak, moguće je definisati opšti okvir za integraciju blokčejn tehnologija u sisteme elektronske trgovine na osnovu trenutno dostupnih tehnoloških rešenja i saznanja.

Iz navedenog proizlazi da osnovni problem istraživanja predstavlja identifikaciju modela elektronske trgovine zasnovanog na blokčejn tehnologiji koji omogućava efikasno upravljanje troškovima, uz očuvanje funkcionalnosti, sigurnosti i prihvatljivog korisničkog iskustva.

1.4 Ciljevi istraživanja

Opšti cilj ovog istraživanja jeste razvoj modela i dizajna distribuirane platforme za elektronsku trgovinu zasnovane na javnoj blokčejn mreži, koja omogućava efikasno upravljanje troškovima izvršavanja uz očuvanje funkcionalnih zahteva, sigurnosti sistema i prihvatljivog korisničkog iskustva.

Tok i sadržaj rada neposredno su uslovljeni definisanim ciljevima istraživanja, pri čemu se opšti cilj razlaže na sledeće specifične ciljeve:

1. Formalizacija referentnog modela elektronske trgovine kroz identifikaciju ključnih elemenata sistema, radi omogućavanja analize dizajna sistema elektronske trgovine.
2. Analiza savremenih informacionih sistema elektronske trgovine, sa ciljem identifikacije njihovih ograničenja i implicitnih pretpostavki u pogledu poverenja i upravljanja podacima korisnika.
3. Uvođenje i formalizacija pojma decentralizacije kao proširenja klasične sociotehničke paradigme informacionih sistema.

4. Definisanje analitičkog okvira za evaluaciju informacionih sistema, koji uključuje metrike za procenu tehničkih atributa infrastrukture, performansi i operativnih svojstava sistema.
5. Analiza javnih blokčejn sistema kao posebne klase distribuiranih informacionih sistema, u okviru prethodno definisanog analitičkog okvira, sa posebnim osvrtom na uticaj decentralizacije na složenost sistema i ograničenja koja iz toga proizlaze.
6. Ispitivanje decentralizacije, sigurnosti i performansi Eterijum platforme, kao i formalizacija pojma gasa kao osnovne jedinice troškovnog modela za utrošeni računarski napor kroz analizu konsenzusnog i izvršnog sloja.
7. Analiza troškovnog modela skladištenja podataka na Eterijum blokčejnu.
8. Analiza enkodirane reprezentacije entiteta maksimalne veličine jedne skladišne jedinice unutar jedinstvenog identifikatora mape vlasništva u okviru standarda za nezamenljive tokene.
9. Komparativna analiza standarda ERC-721, ERC-721A i ERC-721Psi modifikovanih za rad sa e-kuponima kao reprezentativnim studijskim slučajem, sa fokusom na troškove skladišnih operacija nad entitetima prilikom izvršavanja osnovnih funkcija, uključujući kreiranje, prenos i uništavanje.
10. Analiza mehanizma razrešavanja implicitnog vlasništva u standardima ERC-721A i ERC-721Psi, koji nastaje tokom operacija prenosa i uništavanja tokena, uz izvođenje formule za određivanje troška njegovog razrešavanja u zavisnosti od prvog tokena u grupi.
11. Eksperimentalna evaluacija performansi ovako definisanih e-kupona, uz merenje troškova izvršavanja osnovnih operacija u različitim standardima i scenarijima parametara.
12. Ispitivanje proširenja funkcionalnosti standarda za nezamenljive tokene, uključujući modularni dizajn pametnih ugovora i integraciju sa spoljašnjim izvorima podataka, u cilju projektovanja potpuno decentralizovane poslovne logike sistema elektronske trgovine na javnoj blokčejn mreži.
13. Ispitivanje IPFS mreže kao decentralizovanog sloja za skladištenje podataka i prezentacionog sloja radi ostvarivanja potpune decentralizacije aplikacije.

Ovako definisani ciljevi omogućavaju projektovanje arhitektonskog modela distribuirane platforme za elektronsku trgovinu zasnovane na javnoj blokčejn mreži, sa posebnim fokusom na optimizaciju transakcionih naknada korisnika prilikom izvršavanja osnovnih operacija nad entitetima sa jedinstvenim identifikatorima, uključujući njihovo kreiranje, prenos i uništavanje.

Analitički i eksperimentalni okvir uspostavljen kroz ovako definisane ciljeve predstavlja osnovu za sistematsku evaluaciju performansi i troškova predloženog rešenja, kao i za njegovu primenu i dalja unapređenja na drugim blokčejn platformama.

1.5 Hipoteze istraživanja

U okviru ovog istraživanja postavljena je opšta hipoteza koja glasi:

Moguće je dizajnirati distribuiranu platformu elektronske trgovine zasnovanu na javnoj blokčejn mreži, koja primenom optimizovanih modela skladištenja i obrade podataka u

pametnim ugovorima omogućava značajno smanjenje transakcionih troškova korisnika, uz očuvanje funkcionalnih i bezbednosnih svojstava sistema.

Iz ovako definisane opšte hipoteze proizlaze posebne hipoteze koje detaljnije razrađuju pojedinačne aspekte posmatranog problema, kroz formalizaciju ključnih svojstava blokčejn sistema, njihovih ograničenja i mogućnosti primene u elektronskoj trgovini:

1. Javni blokčejn sistemi sa podrškom pametnih ugovora mogu formalno zadovoljiti referentni model elektronske trgovine.
2. Decentralizacija informacionih sistema uvodi dodatnu složenost i zahteva specifične mehanizme koordinacije.
3. Troškovni model zasnovan na gasu omogućava kvantifikaciju računarskog napora prilikom izvršavanja operacija e-trgovine.
4. Optimizacija reprezentacije entiteta primenom standarda za nezamenljive tokene značajno smanjuje troškove gasa.
5. Model implicitnog vlasništva uvodi promenljive troškove u zavisnosti od načina razrešavanja vlasništva.
6. Proširenje funkcionalnosti pametnih ugovora povećava troškove na nelinearan i teško predvidiv način.
7. Primenom IPFS mreže moguće je realizovati decentralizovan sloj za skladištenje podataka i prezentacioni sloj platforme za elektronsku trgovinu, uz očuvanje funkcionalnosti i decentralizovanosti sistema.

Pojedinačne hipoteze koje su dalje razmatrane u disertaciji odnose se na kvantitativnu analizu troškova izvršavanja osnovnih operacija modifikovanih standarda za nezamenljive tokene za rad sa reprezentovanim entitetima u okviru jedne skladišne jedinice:

1. Trošak operacije grupnog kreiranja u standardima sa implicitnim vlasništvom je znatno manji nego u standardu ERC-721.
2. Trošak razrešavanja implicitnog vlasništva zavisi od udaljenosti od prvog eksplicitno definisanog tokena u grupi.
3. Troškovi operacija prenosa i uništavanja u standardima sa implicitnim vlasništvom su znatno veći nego u standardu ERC-721 zbog eksplicitnog razrešavanja vlasništva.
4. Trošak operacije prenosa u standardu ERC-721Psi postaje manji od troška iste operacije u standardu ERC-721A nakon određenog rastojanja između prvog tokena u grupi i posmatranog tokena.
5. Trošak operacija uništavanja samo jednog tokena u transakciji u standardu ERC-721 ostaje konstantan i najmanji je jer jedini omogućava eksplicitan povraćaj gasa usled oslobađanja skladišne jedinice.
6. Trošak uništavanja više tokena u jednoj transakciji u standardu ERC-721Psi počevši od određenog broja uništenih kupona postaje troškovno najpovoljniji od svih standarda.

7. Trošak razrešavanja implicitnog vlasništva u standardu ERC-721A teži vrednosti približno 256 puta većoj od odgovarajućeg troška u standardu ERC-721Psi kada broj kupona kreiranih u jednoj transakciji teži beskonačnosti.
8. Trošak postavljanja pametnog ugovora koji implementira ERC-721Psi standard je najveći u odnosu na ostale posmatrane standarde.

1.6 Metodološki pristup

Metodologija istraživanja zasniva se na kombinaciji teorijsko-analitičkog, konstruktivnog i eksperimentalnog pristupa, sa ciljem sistematske evaluacije mogućnosti primene blokčejn tehnologija u razvoju platformi elektronske trgovine.

Validacija postavljenih hipoteza sprovodi se u skladu sa njihovom prirodom. Posebne hipoteze koje se odnose na formalna svojstva sistema i konceptualne modele validiraju se teorijsko-analitičkim pristupom. Hipoteze koje podrazumevaju dizajn i implementaciju sistema validiraju se konstruktivno, kroz demonstraciju izvodljivosti predloženih rešenja. Pojedinačne hipoteze, koje se odnose na kvantitativne aspekte troškova izvršavanja, validiraju se eksperimentalno, na osnovu merenja potrošnje gasa i komparativne analize dobijenih rezultata.

Teorijsko-analitički pristup obuhvata formalizaciju referentnog modela elektronske trgovine, kao i analitički okvir za evaluaciju informatičkih sistema.

U tom kontekstu analiziraju se osnovna svojstva javnih blokčejn sistema kao specifične klase distribuiranih sistema, koji omogućavaju korišćenje već uspostavljene mrežne i bezbednosne infrastrukture. Analiza obuhvata ključne aspekte kao što su decentralizacija, sigurnost i performanse, uz poseban osvrt na uticaj ovih svojstava na složenost sistema i mehanizme koordinacije.

Fokus istraživanja, kao osnove za primenu konstruktivnog pristupa reprezentaciji entiteta i operacija nad njima, usmeren je na analizu ključnih aspekata Eterijum mreže kroz njen konsenzusni i izvršni sloj.

Dodatno, analiziraju se mogućnosti primene sadržajno adresiranih mreža za decentralizovano skladištenje podataka i realizaciju prezentacionog sloja decentralizovanih aplikacija. U tom kontekstu razmatra se IPFS kao reprezentativan primer infrastrukture koja omogućava dugoročnu dostupnost podataka, očuvanje suvereniteta korisnika nad kriptografskim identitetom i verifikabilnu interakciju sa sistemom.

Konstruktivni pristup koristi se za projektovanje arhitektonskog modela distribuirane platforme elektronske trgovine.

Razvija se model reprezentacije entiteta zasnovan na standardima za nezamenljive tokene u okviru ograničenja Eterijum mreže, pri čemu elektronski kuponi predstavljaju reprezentativan studijski slučaj.

Takođe se u okviru ovog pristupa razmatra modularni dizajn pametnih ugovora kojim se omogućava proširenje funkcionalnosti neophodnih za potpuno ostvarivanje poslovnih zahteva sistema elektronske trgovine na Eterijum blokčejnu.

Eksperimentalni pristup primenjuje se u cilju kvantitativne evaluacije transakcionih troškova korisnika prilikom izvršavanja osnovnih operacija nad entitetima sa jedinstvenim identifikatorima, kroz analizu troškovnog modela zasnovanog na gasu u okviru Eterijum platforme.

U okviru eksperimentalne analize implementirana je enkodirana reprezentacija entiteta elektronske trgovine unutar jedne skladišne jedinice u okviru standarda za nezamenljive tokene, uključujući standarde ERC-721, ERC-721A i ERC-721Psi, pri čemu se vrši poređenje troškova operacija kreiranja, prenosa i uništavanja tokena u kontrolisanim uslovima, sa posebnim naglaskom na broj i tip pristupa skladištu kao dominantnom faktoru troškova prilikom izvršavanja ovih operacija, koji predstavlja primarni predmet optimizacije.

Preostale instrukcije koje se izvršavaju u okviru ovih osnovnih operacija imaju znatno manji uticaj na ukupnu potrošnju gasa i u daljoj analizi biće aproksimativno modelovane radi pojednostavljenja formalnog zapisa.

Kod standarda koji koriste implicitni model vlasništva, posebna pažnja posvećena je analizi algoritama za njegovo razrešavanje. Ovi algoritmi se formalno modeluju, a njihova složenost i trošak izražavaju se kroz aproksimativne funkcije u zavisnosti od rastojanja između posmatranog tokena i prvog tokena u grupnom kreiranju.

Eksperimenti su sprovedeni u kontrolisanom okruženju, uz korišćenje istih ulaznih parametara i uslova izvršavanja za sve posmatrane standarde, kako bi se obezbedila uporedivost rezultata.

Na ovaj način uspostavlja se integrisan metodološki okvir koji omogućava sveobuhvatnu evaluaciju predloženog modela kroz kombinaciju formalne analize, dizajna sistema i empirijskih rezultata.

Predloženi metodološki okvir može se integrisati u šire studije koje se bave ostalim domenima primene informacionih sistema. Kod njih u slučajevima kada je potrebno koristiti više skladišnih jedinica za zapis složenijih struktura podataka, predstavljeni šematski model omogućava relativno jednostavno proširenje analize i kvantifikaciju efikasnosti takvog pristupa.

Dodatno, iako je analiza sprovedena nad javnim blokčejn sistemima, koncept optimalnog pakovanja podataka nije ograničen isključivo na njih. Pristup je primenljiv i u dozvolama ograničenim blokčejn mrežama, jer se zasniva na opštem principu racionalnog korišćenja skladišnog prostora. Ovaj pristup doprinosi smanjenju ukupne količine podataka koju čuvaju čvorovi mreže, smanjenju veličine blokova i povećanju ekonomske održivosti celokupnog ekosistema.

1.7 Eksperimentalni metod

Eksperimentalni metod zasniva se na kvantitativnoj analizi potrošnje gasa tokom izvršavanja ključnih operacija nad entitetima elektronske trgovine. Za merenje potrošnje gasa korišćen je razvojni okvir *Foundry*, koji predstavlja široko prihvaćen alat za razvoj i testiranje pametnih ugovora, čime je obezbeđena determinističnost i ponovljivost rezultata. Ovaj razvojni okvir obuhvata alate:

1. Alat *forge* (v0.2.0), koji se koristi za izgradnju, testiranje i implementaciju pametnih ugovora, kao i za izvršavanje osnovnih operacija i očitavanje vrednosti potrošenog gasa po transakciji.
2. Alat *anvil* (v0.2.0), koji obezbeđuje lokalni čvor Eterijum testne mreže i koristi se za sprovođenje eksperimenata, čime je eliminisan uticaj spoljašnjih faktora.

U okviru ovog okruženja implementirani su pametni ugovori enkodirane reprezentacije elektronskih kupona za sve posmatrane standarde, pri čemu je merena potrošnja gasa za izvršavanje operacija nad elektronskim kuponima.

U cilju preciznog poređenja efikasnosti implementacija, analizirana je isključivo potrošnja gasa posmatranih operacija nad kuponima, bez uključivanja osnovnog troška transakcije od 21.000 jedinica gasa, koji je konstantan i nezavisan od implementacije standarda. U svim modifikovanim standardima pre sprovođenja eksperimenata izvršena je inicijalizacija svih relevantnih internih promenljivih stanja, uključujući promenljivu za identifikaciju narednog tokena za kreiranje, promenljivu za praćenje ukupno uništenih tokena kod ERC-721A standarda, kao i obe bitmape kod ERC-721Psi standarda.

U eksperimentalnom delu analizirana je zavisnost potrošnje gasa od broja kupona kreiranih u jednoj transakciji. Testovi su sprovedeni pozivanjem funkcije za kreiranje sa različitim vrednostima parametra $n \in \{1, 2, 3, 5, 10, 20\}$, pri čemu je za svaku vrednost izmerena ukupna potrošnja gasa. Važno je napomenuti da je broj kreiranih kupona jedini parametar koji utiče na potrošnju gasa ove operacije, dok ostali parametri imaju konstantan uticaj.

Analogno tome, za analizu operacija prenosa i uništavanja korišćen je identifikator kupona oblika $n + 1$, koji predstavlja rastojanje između prvog tokena u grupi i posmatranog tokena. Na ovaj način simuliran je najgori slučaj određivanja implicitnog vlasništva u okviru grupe. U slučaju analize uništavanja više kupona u jednoj transakciji, funkcija uništavanja pozivana je uzastopno za različite identifikatore kupona, čime je obezbeđeno da se u svakom pozivu obrađuje različit kupon.

Svako merenje izvršeno je više puta, a prikazane vrednosti predstavljaju aritmetičke sredine dobijenih rezultata. Dobijeni rezultati upoređeni su između navedenih standarda, pri čemu je referentna tačka bila implementacija zasnovana na ERC-721 modelu. Na osnovu ove komparativne analize izvedeni su zaključci o efikasnosti predloženih pristupa optimizaciji.

Radi obezbeđivanja doslednosti i uporedivosti rezultata, svi pametni ugovori kompajlirani su korišćenjem kompajlera *solc* v0.8.22, sa onemogućenom optimizacijom. Za implementaciju ERC-721 standarda korišćena je referentna implementacija biblioteke *OpenZeppelin* v4.9.0, dok su za standarde ERC-721A i ERC-721Psi korišćene implementacije v4.2.3, odnosno v0.7.0.

Za svaki od posmatranih standarda implementirane su osnovne funkcije za kreiranje, prenos i uništavanje tokena, nakon čega je izvršeno sistematsko merenje potrošnje gasa pri različitim vrednostima ulaznih parametara. Posebno su analizirani:

1. Potrošnja gasa navedenih NFT standarda prilikom višestrukog kreiranja kupona u jednoj transakciji.
2. Potrošnja gasa navedenih NFT standarda prilikom prenosa kupona u zavisnosti od udaljenosti od početnog elementa grupe.
3. Potrošnja gasa navedenih NFT standarda prilikom uništavanja kupona u zavisnosti od udaljenosti od početnog elementa grupe.
4. Potrošnja gasa navedenih NFT standarda prilikom višestrukog uništavanja kupona u jednoj transakciji.
5. Komparativna analiza potrošnje gasa standarda ERC-721A i ERC-721Psi prilikom velikih grupnih operacija.
6. Potrošnja gasa prilikom postavljanja pametnih ugovora za navedene NFT standarde.

1.8 Ograničenja predložene metodologije

U okviru predložene metodologije, za reprezentaciju proizvoda ili usluga sa jedinstvenim identifikatorom na javnim EVM-kompatibilnim blokčejn mrežama korišćeni su postojeći standardi za nezamenljive tokene, modifikovani za potrebe ovog istraživanja, dok standardi za zamenljive tokene koji predstavljaju grupne identifikatore nisu razmatrani.

Kada je reč o metodologiji reprezentacije entiteta u okviru standarda za nezamenljive tokene, optimizacija je ostvarena iskorišćavanjem neiskorišćenih bitova unutar mapiranja vlasništva, čime je omogućeno pakovanje dodatnih atributa proizvoda ili usluge u okviru jedne skladišne jedinice. Na taj način smanjen je broj operacija čitanja i upisa u skladište, što direktno utiče na smanjenje potrošnje gasa. Međutim, efikasnost predloženog pristupa direktno zavisi od trenutnog modela obračuna gasa, posebno odnosa između SLOAD, SSTORE i aritmetičkih operacija. Promene u protokolu naplate skladišnih operacija mogu smanjiti relativnu prednost ove optimizacije.

Prilikom pakovanja više promenljivih u jednu skladišnu jedinicu polazi se od pretpostavke da je veličina adrese u EVM okruženju 160 bita. S obzirom na dinamičan razvoj blokčejn tehnologije, ne postoji garancija da će ova veličina ostati nepromenjena. Eventualna promena mogla bi smanjiti raspoloživi prostor unutar skladišnog slot-a, čime bi bilo neophodno prilagoditi predloženi model optimizacije, pri čemu bi osnovni koncept metodologije ostao nepromenjen.

Upotreba implicitnog vlasništva uvodi ograničenja u kontekstu grupnog kreiranja NFT-ova. Kada se više tokena kreira u okviru iste transakcije, podaci se eksplicitno čuvaju samo za prvi token, dok ostali implicitno nasleđuju iste vrednosti. Ovakav pristup je efikasan u slučaju identičnih instanci, ali postaje ograničavajući u scenarijima koji zahtevaju veću raznovrsnost atributa unutar jedne serije.

Analiza je fokusirana na statičke modele proizvoda i usluga, kod kojih se atributi ne menjaju često ili ne zahtevaju složene dinamičke interakcije. Poslovni modeli sa izraženom dinamikom podataka nisu obuhvaćeni ovom analizom, a njihova podrška bi zahtevala dodatne mehanizme ažuriranja i potencijalno drugačiju strukturu skladištenja.

Pre dizajna sistema za elektronsku trgovinu neophodno je pažljivo proceniti stepen potrebne nadogradivosti i modularnosti, kao i njihov odnos prema složenosti implementacije i bezbednosnim rizicima. Svaka od navedenih arhitektonskih odluka ima direktan uticaj na potrošnju gasa, veličinu bajtkoda i ukupne troškove postavljanja i održavanja platforme, kao i na troškove izvršavanja transakcija krajnjih korisnika, što nije eksplicitno analizirano u okviru predložene metodologije zbog složenosti sistema.

Dodatno, u okviru predložene metodologije nije detaljno analizirana bezbednost integracije blokčejn sistema sa spoljašnjim izvorima podataka, niti je detaljno analiziran uticaj predloženih rešenja na performanse sistema. Za predloženi sloj skladištenja podataka i prezentacioni sloj nije realizovana potpuna implementacija niti eksperimentalna evaluacija, budući da je fokus disertacije usmeren na realizaciju i izvršavanje korisničkih transakcija platforme za elektronsku trgovinu. Takođe, nisu sprovedena eksperimentalna merenja performansi pristupa sadržaju i vremena odziva, iako ovi parametri imaju značajan uticaj na upotrebljivost platforme iz perspektive krajnjih korisnika.

Predstavljeni koncept može se primeniti i na druge, ne-EVM blokčejn sisteme, uz ograničenje da se razmatraju blokčejn arhitekture zasnovane na linearnom nizu blokova povezanih u strogo hronološki lanac, dok paradigme zasnovane na usmerenim acikličnim grafovima nisu obuhvaćene ovim istraživanjem.

U ovom slučaju dodatno ograničenje predstavlja pretpostavka da izabrani blokčejn sistem podržava postojanje mape vlasništva u okviru jednog pametnog ugovora, kao što je slučaj sa Eterijum

blokčejnom. Ovakav model podrazumeva mogućnost skladištenja svih tokena unutar jednog ugovora sa praktično neograničenim skladišnim kapacitetom, čime se omogućava primena identične NFT logike.

Suprotan primer predstavlja arhitektura kakva je primenjena na asinhronoj blokčejn mreži *Telegram Open Network*, gde se svaki NFT implementira kao zaseban pametni ugovor, sa ograničenim skladišnim kapacitetom i uslovljenom asinhronom komunikacijom. U takvom modelu nije moguće primeniti model implicitnog vlasništva na način korišćen u analiziranim NFT standardima.

1.9 Naučni doprinos

Naučni doprinos ove disertacije ogleda se u sistematskom povezivanju teorijskih, konstruktivnih i eksperimentalnih aspekata primene blokčejn tehnologija u domenu elektronske trgovine, kroz razvoj integrisanog modela koji omogućava kvantitativno i kvalitativno sagledavanje ograničenja i mogućnosti ovakvih sistema. Na osnovu sprovedenog istraživanja i dobijenih rezultata, izdvajaju se sledeći naučni doprinosi:

1. Definisanje referentnog modela elektronske trgovine u kontekstu javnih blokčejn sistema, čime se uspostavlja jasan teorijski okvir za analizu i poređenje različitih arhitektonskih pristupa.
2. Definisanje analitičkog okvira za evaluaciju blokčejn sistema kao posebne klase distribuiranih informacionih sistema, kojim se omogućava strukturisano razmatranje odnosa između decentralizacije, sigurnosti i performansi, kao i njihovog uticaja na složenost i operativna svojstva sistema.
3. Formalizacija optimizovanog modela reprezentacije entiteta elektronske trgovine sa jedinstvenim identifikatorima unutar jedne skladišne jedinice, zasnovanog na standardima za nezamenljive tokene, čime se smanjuje broj pristupa skladištu kao dominantnog faktora troškova.
4. Formalna analiza i modelovanje mehanizama implicitnog vlasništva u standardima za nezamenljive tokene, uključujući izvođenje aproksimativnih funkcija troška eksplicitnog razrešavanja vlasništva u zavisnosti od udaljenosti od početnog elementa grupe.
5. Komparativna analiza troškovne efikasnosti osnovnih operacija standarda ERC-721, ERC-721A i ERC-721Psi u različitim operativnim scenarijima, zasnovana na eksperimentalnim merenjima potrošnje gasa.
6. Predlog arhitektonskog modela decentralizovane platforme za elektronsku trgovinu zasnovane na javnoj blokčejn mreži i IPFS mreži, koji integriše optimizovane modele skladištenja podataka, modularni dizajn pametnih ugovora, kao i sloj za skladištenje podataka i prezentacioni sloj zasnovani na sadržajnom adresiranju, uz očuvanje suvereniteta korisnika nad kriptografskim identitetom.

1.10 Struktura disertacije

Rad je strukturiran kroz više međusobno povezanih celina koje prate logički tok od teorijskih osnova, preko analitičkog i metodološkog okvira, do konstruktivne realizacije i eksperimentalne evaluacije predloženog rešenja, pri čemu se u uvodnom poglavlju sistematski definišu predmet, ciljevi, hipoteze i metodološki pristup istraživanja.

U drugoj glavi uspostavlja se konceptualni okvir za analizu savremenih informacionih sistema. Najpre se razmatra odnos radnih i informacionih sistema, kao i formalna izražajnost poslovnih zahteva koje takvi sistemi treba da podrže. Zatim se uvodi informacioni sistem elektronske trgovine kao reprezentativan primer savremenog poslovnog informacionog sistema, kroz analizu njegovih osnovnih karakteristika, referentnog modela i tehnoloških zahteva. Dalje se analiziraju osnovni arhitektonski obrasci organizacije informacionih sistema, uz naglasak da većina savremenih rešenja zadržava centralizovan model upravljanja, nezavisno od toga da li su implementirana na jednom ili više računarskih čvorova, kao i ograničenja takvog pristupa. U tom kontekstu uvodi se pojam decentralizacije kroz različite dimenzije, kao ključni teorijski okvir za razumevanje potencijala unapređenja autonomije korisnika i raspodele kontrole nad sistemom.

Treća glava uspostavlja analitički okvir za evaluaciju informacionih sistema kroz njihova tehnička svojstva i kompromis između performansi, skalabilnosti, dostupnosti i bezbednosti. Analiza obuhvata fundamentalne principe i ograničenja distribuiranih sistema, uključujući CAP teorem, skalabilnost, zagušenje, režijsko opterećenje, koordinaciju, propagaciju, replikaciju, konsenzus i kontrolu konkurentnog pristupa, kao i njihove posledice po performanse i operativnu stabilnost sistema. Posebna pažnja posvećena je operativnim svojstvima informacionih sistema kroz bezbednosne modele, kriptografske mehanizme, autentikaciju, autorizaciju i evidenciju, čime se pokazuje da tehnička arhitektura sistema istovremeno određuje i njegove mogućnosti i njegova ograničenja.

Četvrta glava rada razmatra blokčejn kao infrastrukturni sloj distribuiranih sistema. Polazeći od teorijskih osnova, uključujući kriptografsko vremensko obeležavanje, dokaz uloženog rada i rane modele elektronskog novca, uvodi se kontekst nastanka savremenih blokčejn mreža. Posebno se analizira Bitcoin kao prvi praktično realizovan sistem, koji kroz javno dostupnu evidenciju transakcija i Nakamoto konsenzus omogućava postizanje saglasnosti u okruženju bez međusobnog poverenja. Dalje se razmatraju ključne arhitektonske komponente blokčejn sistema, uključujući strukturu blokova i transakcija, modele upravljanja stanjem i kontrole pristupa, pametne ugovore, apstrakciju naloga i međulančanu komunikaciju. Posebna pažnja posvećena je blokčejn trilemi između decentralizovanosti, sigurnosti i skalabilnosti, ukorenjenoj u samim teorijskim osnovama blokčejn sistema.

Peta glava pruža pregled arhitekture Eterijum blokčejna kroz analizu konsenzusnog i izvršnog sloja. Detaljno se razmatraju mehanizmi zasnovani na dokazu uloga, uključujući raspodelu validatora, proces glasanja, kao i algoritme izbora kanonske grane i finalizacije, čime se omogućava analiza bezbednosnih mehanizama, ekonomskih modela i potencijalnih pretnji stabilnosti mreže. U nastavku se obrađuju unutrašnja struktura izvršnog sloja sistema, model globalnog stanja, rad Eterijum virtuelne mašine (EVM) i proces obrade transakcija, čime se pruža integrisan i celovit prikaz funkcionisanja Eterijuma kao decentralizovanog računarskog sistema.

Šesta glava rada posvećena je metodološkoj optimizaciji pametnih ugovora za potrebe elektronske trgovine, sa ciljem smanjenja potrošnje gasa i unapređenja korisničkog iskustva. U prethodnom radu razmatrano je modelovanje elektronskih kupona kao specifične instance digitalnih entiteta, što predstavlja reprezentativan studijski slučaj za širu primenu nezamenljivih tokena u elektronskoj trgovini. Analiza je usmerena na operacije pristupa trajnom skladištu kao ključni faktor optimizacije, budući da one dominantno određuju ukupnu cenu izvršavanja na blokčejn sistemima, uz predlog kompaktnog pakovanja više informacija u jednu skladišnu jedinicu i korišćenja postojećih NFT standarda. Imajući u vidu široku primenu Eterijum virtuelne mašine, razmatraju se standardi ERC-721, ERC-721A i ERC-721Psi kroz detaljnu analizu troškova operacija kreiranja, prenosa i uništavanja. Predloženi pristupi evaluirani su eksperimentalnim merenjima potrošnje gasa, čime se omogućava objektivno poređenje njihove efikasnosti i praktične primenljivosti.

Sedma glava posvećena je arhitekturi IPFS mreže i njenoj ulozi u decentralizovanim aplikacijama. Analizirani su modeli sadržajnog adresiranja, organizacije i verifikacije podataka, mehanizmi distribuiranog pronalaženja i razmene sadržaja, kao i mogućnosti pristupa sadržaju putem IPFS pristupnika. Razmotren je problem dugoročne dostupnosti podataka i prikazano je rešenje zasnovano na decentralizovanim ekonomskim podsticajima. Na kraju su analizirani mehanizmi pristupa decentralizovanim aplikacijama putem kriptografskih novčanika i ENS sistema za imenovanje.

Osma glava posvećena je diskusiji o mogućnostima proširenja, unapređenja i praktične primene predložene arhitekture sistema elektronske trgovine zasnovane na blokčejn tehnologijama. Razmatrani su regulatorni, bezbednosni i tehnički aspekti implementacije decentralizovanih sistema, kao i napredne funkcionalnosti NFT standarda i obrasci modularnog dizajna pametnih ugovora. Dodatno su analizirani mehanizmi povezivanja blokčejn sistema sa spoljašnjim izvorima podataka primenom decentralizovanih mreža proročišta i zkTLS pristupa, kao i specifičnosti decentralizovanog skladištenja podataka zasnovanog na IPFS ekosistemu. Poseban akcenat stavljen je na izazove vezane za performanse, privatnost, dugoročnu dostupnost podataka i zavisnost od postojeće internet infrastrukture.

Deveta glava predstavlja završni deo disertacije u kojem su objedinjeni rezultati teorijske analize, konstruktivnog modelovanja i eksperimentalne evaluacije. Razmatra se doprinos istraživanja razumevanju i projektovanju decentralizovanih sistema elektronske trgovine, uz osvrt na mogućnosti daljeg razvoja i unapređenja predložene arhitekture.

Glava 2

Savremeni informacioni sistemi

Ideja *sistema* prisutna je praktično tokom čitave istorije ljudske civilizacije, iako dugo nije bila formalno definisana kao posebna naučna disciplina. Još od najranijih oblika organizovanog društva, trgovine i proizvodnje, ljudi su razvijali različite oblike međusobno povezanih procesa, pravila, resursa i učesnika koji su zajedno činili funkcionalne celine usmerene ka ostvarivanju definisanih ciljeva.

Na primer, još u Mesopotamiji (oko 3000. p.n.e.) postoje zapisi na glinenim tablicama o trgovini žitom, evidenciji dugova, kreditnim odnosima i organizaciji rada, dok Hamurabijev zakonik (oko 1750. p.n.e.) uspostavlja formalna pravila ne samo za poslovne odnose, već i za šire državne i društvene procese, uključujući regulisanje ugovora, odgovornosti, radnih obaveza i sankcija [1].

Tokom XX veka dolazi do formalizacije teorijskih okvira za proučavanje sistema, posebno kroz razvoj teorije opštih sistema [2], čiji je cilj bio identifikovanje univerzalnih principa zajedničkih različitim vrstama sistema. Dalji razvoj teorije sistema doveo je do socio-tehničkog pristupa [3], koji se fokusira na zajedničku optimizaciju društvenih i tehničkih podсистема koji međusobno interaguju i zajednički utiču na funkcionisanje organizacije.

S obzirom na to da informacioni sistemi danas predstavljaju jedan od najzastupljenijih oblika tehnološke podrške poslovnim procesima, u nastavku ovog poglavlja biće analizirano njihovo mesto u okviru radnih sistema [4]. Posebna pažnja biće posvećena načinu na koji savremeni informacioni sistemi funkcionišu, kao i prednostima i ograničenjima koja proizlaze iz dominantnog načina njihove primene u savremenom poslovanju.

2.1 Radni sistem

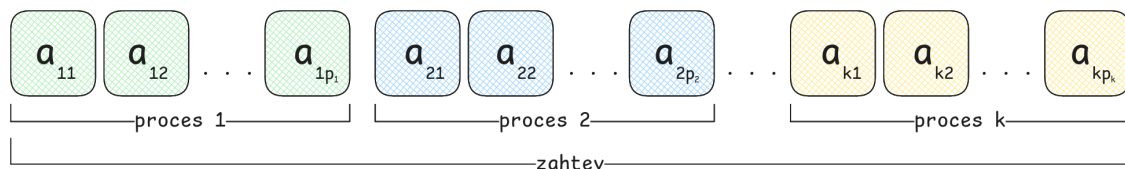
Osnovna ideja radnog sistema je da prikazuje kako se posao u organizaciji zaista odvija u praksi, nezavisno od stepena automatizacije ili nivoa tehnološke podrške.

Definicija 2.1. *Radni sistem (eng. Work system) [5] je sociotehnički sistem u kome ljudski učesnici i/ili mašine obavljaju poslovni proces koristeći informacije, tehnologiju i druge resurse za proizvodnju proizvoda i/ili usluga za interne ili eksterne kupce.*

U okviru ovog pristupa identifikovano je devet osnovnih elemenata koji zajedno čine opšti model radnog sistema [6], a njihova međusobno neraskidiva povezanost doprinosi složenosti radnog sistema. Ovi elementi predstavljaju analitički okvir koji se može primeniti na različite podtipove sistema,

poput informacionih sistema, projekata, lanaca snabdevanja ili sistema elektronske trgovine, dok se u određenim kontekstima pojedini elementi mogu prilagođavati ili izostavljati.

1. **Procesi i aktivnosti** se odvijaju u radnom sistemu kako bi se proizveli proizvodi i usluge za njihove kupce. Radni sistem mora sadržati najmanje jednu aktivnost jer u suprotnom ne izvršava nikakvu funkciju. Mnogi važni radni sistemi obavljaju organizovane aktivnosti koje se u velikoj meri oslanjaju na ljudsko rasuđivanje i improvizaciju.
2. **Učesnici** su ljudi koji obavljaju posao unutar radnog sistema, bez obzira na to da li koriste informacione tehnologije ili ne. U mnogim slučajevima i sami kupci mogu biti učesnici.
3. **Informacije** obuhvataju sve oblike podataka koje radni sistem kreira, koristi, skladišti, prenosi, ažurira ili briše tokom aktivnosti. Ovo ne obuhvata samo digitalne podatke u informacionim tehnologijama, već i verbalne dogovore, dokumenta, implicitno znanje, beleške i sl.
4. **Tehnologije** obuhvataju fizičke i/ili digitalne alate koji pomažu učesnicima da obave posao ili da ga u potpunosti automatizuju. Tehnologije mogu obuhvatati jednostavne fizičke alate, poput čekića, kao i računare, softver i potpuno automatizovane digitalne agente.
5. **Proizvod** sistema rada predstavlja rezultat njegovog funkcionisanja, jer radni sistem kao krajnji izlaz proizvodi određenu informaciju, fizičku stvar i/ili uslugu koja donosi vrednost korisniku. Procena uspešnosti sistema rada može se meriti na osnovu proizvoda.
6. **Kupci** su oni koji dobijaju i koriste proizvode i usluge radnog sistema. Oni mogu biti spoljašnji (kupci organizacije) ili unutrašnji (npr. drugi odsek u kompaniji).
7. **Okruženje** predstavlja spoljne faktore koji utiču na rad sistema: zakone i regulativu, tržišnu konkurenciju, kulturu, standarde, organizacione politike, istoriju firme i sl. Ignorisanje uticaja okruženja može dovesti do pogrešne analize i neuspeha prilikom dizajna sistema.
8. **Infrastruktura** obuhvata resurse na koje se radni sistem oslanja i koristi, ali koji nisu direktno pod njegovom kontrolom. To mogu biti drugi sistemi, ljudska infrastruktura, zajedničke baze podataka, „platforme u oblaku” i sl.
9. **Strategije** određuju pravac i nameru. One postoje na nivou preduzeća, odeljenja i konkretnog sistema rada. Strategije moraju biti usklađene i koherentne, jer neusaglašene strategije dovode do neuspešne implementacije i pogrešnih prioriteta.



Slika 2.1: Prikaz jednog zahteva kroz njemu pridružene procese i aktivnosti.

Procesi i aktivnosti predstavljaju konkretan rad koji se obavlja unutar radnog sistema, pri čemu se njihovo izvršavanje definiše kroz formalne i/ili neformalne zahteve. Jedan proces obuhvata jednu ili više aktivnosti, dok pojedinačni radni zahtev može obuhvatati jedan ili više procesa, pri čemu

se kroz procese i aktivnosti operacionalizuju poslovne potrebe, pravila, izuzeci, ograničenja i ciljevi koji zajedno čine sadržaj zahteva.

Na ovaj način omogućava se sveobuhvatna analiza, razumevanje, objektivno vrednovanje i unapređenje poslovnih aktivnosti, pri čemu se zahtevi razlažu na merljive i proverljive kriterijume performansi i kvaliteta, čime se obezbeđuje njihova operacionalizacija i mogućnost formalnog modelovanja i transformacije u izvršive modele.

Radni sistemi postoje kako bi stvarali vrednost za svoje korisnike, pa se njihova uspešnost ocenjuje kako prema efikasnosti unutrašnjih procesa, tako i prema percepciji i zadovoljstvu korisnika proizvodima i uslugama koje sistem generiše, odnosno *korisničkom iskustvu*. U zavisnosti od svrhe i konteksta, pojam *vrednosti* može se interpretirati na različite načine, pa radni sistemi mogu da se razmatraju kroz četiri osnovne kategorije:

1. Poslovni sistemi usmereni su na stvaranje ekonomske vrednosti za klijente i organizaciju [7],
2. Javni sistemi usmereni su na stvaranje društvene vrednosti kroz pružanje usluga građanima i unapređenje javnog interesa [8],
3. Neprofitni sistemi usmereni su na stvaranje zajedničkog dobra bez cilja ostvarivanja profita [9],
4. Lični radni sistemi odnose se na načine na koje pojedinci koriste informacije i tehnologije za postizanje ličnih ciljeva i povećanje lične produktivnosti.

Bez obzira na kategoriju, svi radni sistemi dele zajednički ontološki minimum, odnosno procese i aktivnosti koje učesnici obavljaju koristeći informacije i tehnologije radi stvaranja određene vrednosti za jasno definisane korisnike.

Među različitim tipovima radnih sistema, informacioni sistemi predstavljaju poseban slučaj u kojem su procesi i aktivnosti u značajnoj meri podržani ili automatizovani informacionim tehnologijama. Zbog toga će u daljem radu fokus biti na informacionim sistemima kao izvršivim modelima radnih sistema.

2.2 Informacioni sistem

Da bi se precizno definisao informacioni sistem, potrebno je najpre razjasniti pojam informacionih tehnologija, koje se zasnivaju na skupu osnovnih tehničkih komponenti [10]. U svom osnovnom obliku, informacioni sistemi se zasnivaju na dve fundamentalne komponente:

1. **Hardver**, koji obezbeđuje fizičku infrastrukturu neophodnu za izvršavanje računarskih procesa,
2. **Softver**, koji predstavlja logički sloj sistema i definiše njegove funkcionalnosti, ponašanje i stepen automatizacije poslovnih procesa.

Međutim, u savremenim informacionim sistemima, radi preciznije analize i jasnijeg modelovanja, ove osnovne komponente se u literaturi često posmatraju kroz dodatna dva konceptualna sloja, koji u sebi implicitno obuhvataju i hardverske i softverske elemente:

3. **Bazu podataka**, koja obezbeđuje mehanizme trajnog čuvanja podataka, kao i logičke strukture i softverske servise za njihovo organizovano čitanje, pisanje i upravljanje,

4. **Mreže**, kao infrastrukturni i komunikacioni sloj koji omogućava razmenu podataka, koordinaciju i interoperabilnost između udaljenih delova sistema.

Definicija 2.2. *Informacione tehnologije (IT) predstavljaju skup tehničkih sredstava, uključujući hardver, softver, baze podataka i mreže, namenjenih obradi, čuvanju, razmeni i prikazu informacija neophodnih za realizaciju poslovnih procesa i aktivnosti.*

Ovakva četvorokomponentna podela ne uvodi nove fundamentalne tehničke osnove sistema, već predstavlja analitički okvir koji omogućava jasnije razdvajanje odgovornosti, lakše razumevanje arhitekture i precizniju analizu ponašanja savremenih informacionih sistema.

Definicija 2.3. *Informacioni sistem predstavlja sociotehnički sistem koji nastaje integracijom radnog sistema sa informacionim tehnologijama radi podrške ili automatizacije procesa i aktivnosti u okviru tog sistema.*

Bez obzira na arhitekturu, veličinu ili primenjenu tehnologiju, svaki informacioni sistem obuhvata osnovne funkcionalne: ulaz podataka, njihovu obradu i generisanje izlaznih informacija [11]. Savremeni informacioni sistemi dodatno uključuju mehanizme skladištenja podataka, kontrole i povratne sprege, čime se omogućava adaptivno i kontinuirano funkcionisanje sistema.

2.2.1 Tjuringova kompletnost

Radni sistem formuliše zahteve u domenu realnih poslovnih procesa, dok informacioni sistem predstavlja njihovu formalizovanu i izvršivu reprezentaciju u digitalnom okruženju. Preslikavanje zahteva radnog sistema u formalni domen informacionog sistema ne predstavlja trivijalan proces, već podrazumeva interpretaciju, razlaganje, preciziranje i formalizaciju poslovnih zahteva.

Samo oni zahtevi koji se mogu dovoljno precizno formalizovati, operacionalizovati i algoritamski izraziti mogu biti reprezentovani i izvršeni u okviru formalnog jezika informacionog sistema. Proces njihove transformacije obuhvata identifikaciju, analizu, specifikaciju, validaciju i upravljanje zahtevima tokom celokupnog životnog ciklusa sistema [12].

Definicija 2.4. *Programski jezik [13] predstavlja formalni jezik namenjen opisivanju računarskih procesa, čija sintaksa i semantika definišu način reprezentacije podataka, operacija i kontrolnih struktura koje se mogu jednoznačno interpretirati i izvršiti od strane računarskog sistema.*

U sistemima projektovanim za unapred definisan i konačno ograničen domen zahteva, upotreba ograničenih programskih jezika predstavlja sasvim dovoljan pristup. Međutim, u slučajevima kada buduća struktura procesa, poslovnih pravila i njihovih kombinacija nije unapred poznata, odnosno kada je domen zahteva strukturno neformalan, svaka pojava nove poslovne logike zahtevala bi re-dizajn informacionog sistema ukoliko programski jezici koje koristi poseduju ograničenu izražajnu moć.

Definicija 2.5. *Tjuringova kompletnost [14] predstavlja fundamentalno svojstvo programskog jezika koje omogućava izražavanje proizvoljnih algoritama uz dostupnost osnovnih kontrolnih struktura kao što su sekvencijalno izvršavanje, uslovno grananje i ponavljanje izvršavanja instrukcija.*

Tjuringova kompletnost programskog jezika informacionog sistema predstavlja neophodan uslov u situacijama u kojima radni sistem ne može unapred ograničiti oblik svojih budućih zahteva, budući da omogućava implementaciju novih tipova funkcionalnih zahteva čija se struktura i logika ne

moгу predvideti u fazi projektovanja sistema, bez potrebe za redizajnom osnovne arhitekture informacionog sistema. Jezici koji podržavaju Tjuringovu kompletnost se nazivaju *Tjuring kompletni jezici*.

Iako Tjuringova kompletnost obezbeđuje maksimalnu izražajnost, ona istovremeno uvodi povećanu složenost u verifikaciji, analizi i kontroli ponašanja sistema. Zbog toga se u pojedinim domenima namerno koriste netjuring kompletni jezici kako bi se obezbedila veća sigurnost, predvidivost i formalna proverljivost izvršavanja.

2.3 Informacioni sistem elektronske trgovine

Kako se informacioni sistemi mogu posmatrati kao tehnološke implementacije radnih sistema različitih namena, poslovni informacioni sistemi nastaju primenom informacionih sistema u domenu poslovnih procesa [15]. Većina savremenih informacionih sistema pripada kategoriji poslovnih informacionih sistema, koji predstavljaju najširi okvir digitalne podrške poslovnim aktivnostima organizacije.

U okviru poslovnih informacionih sistema, digitalna trgovina (eng. *digital commerce*) predstavlja specijalizovanu oblast usmerenu na upravljanje celokupnim digitalnim korisničkim iskustvom kroz različite digitalne kanale, dok elektronska trgovina (eng. *e-commerce*) predstavlja užu podskup digitalne trgovine fokusiran prvenstveno na elektronsko izvršavanje kupoprodajnih transakcija.

Definicija 2.6. *Elektronska trgovina [16] predstavlja prodaju ili kupovinu robe i usluga realizovanu putem računarskih mreža korišćenjem metoda posebno projektovanih za prijem ili izdavanje porudžbina.*

E-trgovina predstavlja jedan od najdinamičnijih i najvrednijih segmenata savremenih informacionih sistema. Prema savremenim empirijskim analizama digitalne ekonomije [17], industrija globalne e-trgovine procenjena je na približno 21,1 bilion američkih dolara u 2023. godini, uz projekcije snažnog dugoročnog rasta u narednoj deceniji, prema tržišnim analizama na 183,8 biliona do 2032. godine, što potvrđuje njenu izuzetnu ekonomsku i tehnološku relevantnost.

Ovakav obim i složenost digitalne trgovine ukazuju na potrebu za sistematičnim modelovanjem i struktuiranjem njenog domena kako bi se omogućila analiza, klasifikacija i diskusija o sistemima elektronske trgovine. U tu svrhu koristi se referentni model elektronske trgovine [18], koji obuhvata tri međusobno povezana podmodela:

1. **Poslovni model**, koji definiše vrstu poslovanja, proizvode i usluge koje sistem elektronske trgovine nudi korisnicima, kao i mehanizme ostvarivanja prihoda.
2. **Model ponašanja kupaca**, koji se bavi razumevanjem i modelovanjem obrazaca interakcije korisnika sa sistemom putem njegovog prezentacionog sloja, uključujući generisanje zahteva, navigaciju i korišćenje dostupnih funkcionalnosti.
3. **Model resursa informacione tehnologije**, koji obuhvata infrastrukturu i tehnološke resurse čije korišćenje i održavanje omogućavaju realizaciju i podršku sistema elektronske trgovine.

Proizvodi i usluge u sistemima elektronske trgovine predstavljaju poslovne entitete koji moraju posedovati jedinstvenu identifikaciju radi omogućavanja pretrage, razmene podataka, upravljanja

životnim ciklusom i interoperabilnosti između različitih informacionih sistema. Iako konkretni mehanizmi identifikacije mogu varirati u zavisnosti od domena primene, postojanje identifikatora predstavlja neophodan uslov za upravljanje katalogom proizvoda, realizaciju porudžbina i izvršavanje transakcija [19].

Budući da korisnici predstavljaju osnovni izvor prihoda elektronske trgovine, degradacija performansi može dovesti do napuštanja pojedinačnih sesija, smanjenja stope konverzije i dugoročnog gubitka korisnika. Empirijska istraživanja pokazuju da čak i relativno mala povećanja vremena odziva i složenosti interakcije mogu dovesti do povećane stope napuštanja sajta, smanjenog zadovoljstva korisnika i manjeg broja realizovanih kupovina [20, 21].

Posledica. *Korisničko iskustvo u sistemima elektronske trgovine direktno zavisi od performansi informacione infrastrukture na kojoj se sistem zasniva.*

Tehnološki zahtevi platformi za e-trgovinu često predstavljaju referentni nivo zahteva za savremene poslovne informacione sisteme, jer informacioni sistemi u drugim domenima, poput javnog sektora, neprofitnih organizacija ili ličnih sistema, razlikuju se prvenstveno po svojim ciljevima i korisničkim zahtevima. Međutim, osnovni tehnološki izazovi i principi u velikoj meri ostaju isti.

U daljem izlaganju, elektronska trgovina se koristi kao dominantni referentni primer savremenog informacionog sistema, jer objedinjuje tipične izazove visokog obima obrade podataka, kompleksnih sistemskih arhitektura i upravljanja stanjem sistema. Shodno tome, svi naredni koncepti i analize biće predstavljeni bez eksplicitnog navođenja domena primene, podrazumevajući kontekst e-trgovine. Međutim, razmatrani principi nisu ograničeni na ovaj domen, već predstavljaju opšte obrasce i mehanizme primenljive na širi spektar informacionih sistema.

2.4 Arhitektonska organizacija informacionih sistema

Program predstavlja statički, fizički zapis koda na uređaju (hardveru) koji sam po sebi ne izvršava nikakvu aktivnost. Kada se pokrene njegova instanca, tj. kada se program učitava u radnu memoriju računara, nastaje proces. Po završetku izvršavanja, proces se uklanja iz radne memorije i time prestaje da postoji.

Definicija 2.7. *Proces je instanca aplikacije koja ima sopstveni memorijski prostor i kontekst izvršavanja.*

Najčešće se softver izvršava na uređajima koji poseduju operativni sistem [22], čija je uloga da omogućiti istovremeno izvršavanje i upravljanje više procesa na istom uređaju: planiranje njihovog izvršavanja, dodelu resursa, sinhronizaciju i međuprocensku komunikaciju (eng. *Inter-Process Communication, IPC*).

U zavisnosti od fizičke strukture procesora može se u istom trenutku izvršavati jedan ili više procesa, pri čemu operativni sistem „naizmenično” dodeljuje procesor procesima tako što pauzira, čuva i vraća njihov kontekst izvršavanja.

Međutim, softver se može izvršavati i na uređajima bez klasičnog operativnog sistema, u tzv. *bare-metal* okruženju, gde postoji samo jedan proces koji direktno radi na hardveru. Višestruki procesi mogu postojati ukoliko je implementirano posebno izvršno okruženje koje to omogućava.

Definicija 2.8. *Instanca informacionog sistema je skup pokrenutih procesa koji rade u koordinaciji kako bi obezbedili punu funkcionalnost aplikacije.*

Pokrenuti informacioni sistem nije statičan, već se njegovo ponašanje izražava kroz kontinuirano izvršavanje procesa i promene unutrašnjeg stanja sistema. U tom kontekstu uvode se dva ključna pojma:

Definicija 2.9. *Stanje informacionog sistema predstavlja skup svih podataka i sistemskih parametara koji u datom trenutku u potpunosti opisuju njegovu instancu.*

Definicija 2.10. *Napredovanje informacionog sistema predstavlja proces prelaska sistema iz jednog stanja u drugo kao posledicu njegovog izvršavanja i obrade zahteva.*

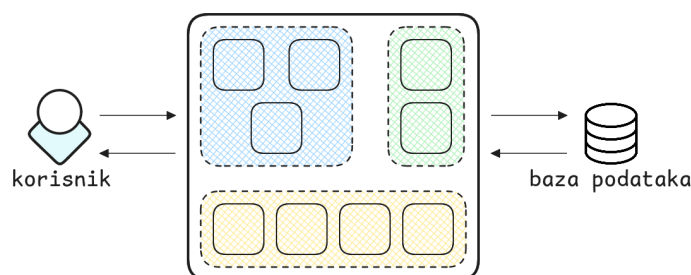
U zavisnosti od načina na koji su procesi i aktivnosti organizovani unutar instance informacionog sistema mogu se identifikovati različiti arhitektonski pristupi. Ovi pristupi određuju strukturu sistema, način raspodele odgovornosti između komponenti, njihove međusobne zavisnosti i mehanizme komunikacije putem kojih sistem ostvaruje svoje funkcionalno ponašanje.

Najjednostavniji oblik organizacije informacionog sistema podrazumeva da se sve funkcionalne komponente izvršavaju na jednom uređaju, pri čemu moduli međusobno komuniciraju pozivima funkcija ili metoda unutar istog procesa.

Monolitna arhitektura [23]. Monolitna arhitektura predstavlja pristup u kome su sve funkcionalne komponente informacionog sistema objedinjene u jedinstvenu aplikacionu celinu. U okviru ovog pristupa korisnički interfejs (ulaz/izlaz), poslovna logika (obrada) i upravljanje podacima (baza podataka) implementirani su kao delovi jednog izvršnog sistema.

Glavna prednost monolitne arhitekture ogleda se u njenoj jednostavnosti, zbog čega je pogodna za manje i srednje aplikacije. Razvoj, testiranje, implementacija i nadgledanje sistema u takvom okruženju su relativno jednostavni jer se sve komponente nalaze u okviru jedinstvene aplikacije.

Međutim, sa rastom obima i složenosti sistema javljaju se poteškoće u održavanju, jer promene u jednom delu sistema mogu imati neočekivane posledice u drugim delovima aplikacije. Zbog toga se u praksi često primenjuje pristup *modularnog monolita*, u kojem se unutar monolitne aplikacije jasno razdvajaju funkcionalni moduli kako bi se olakšalo održavanje i eventualna buduća migracija ka drugim arhitekturama.

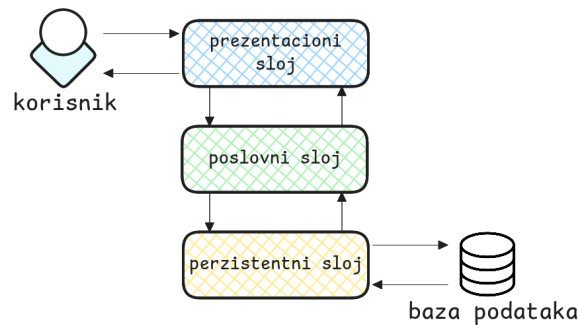


Slika 2.2: Modularni monolit kao organizacioni obrazac monolitne arhitekture.

Višeslojna arhitektura [24]. Višeslojna arhitektura zasniva se na organizaciji sistema u više logičkih slojeva, pri čemu svaki sloj ima jasno definisanu funkciju i odgovornost. Ovakva struktura omogućava razdvajanje različitih aspekata sistema, čime se postižu apstrakcija i enkapsulacija funkcionalnosti.

Komunikacija između slojeva odvija se prema unapred definisanim pravilima, najčešće između susednih slojeva, čime se smanjuje međusobna zavisnost komponenti i olakšava održavanje sistema. Tipičan primer ovakve organizacije obuhvata sledeće slojeve:

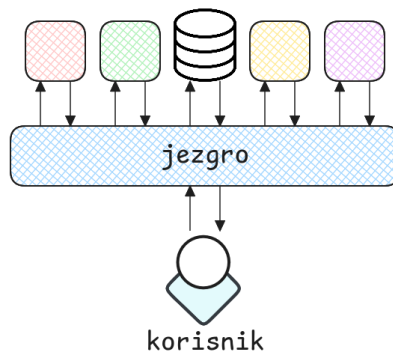
1. *Prezentacioni sloj* - korisnički interfejs koji omogućava komunikaciju korisnika sa sistemom,
2. *Poslovni sloj* - sloj koji implementira poslovnu logiku i pravila rada sistema,
3. *Perzistentni sloj* - sloj koji obezbeđuje preslikavanje objekata iz aplikacionog domena u strukture podataka,
4. *Sloj podataka* - baza podataka ili drugi mehanizmi za trajno skladištenje i pristup podacima.



Slika 2.3: Logički slojevi i njihove međusobne zavisnosti u višeslojnoj arhitekturi.

Mikrokernel arhitektura [25]. Mikrokernel arhitektura zasniva se na razdvajanju osnovne funkcionalnosti sistema od dodatnih modula. Centralni deo sistema čini jezgro (eng. *kernel*) koje obezbeđuje minimalni skup osnovnih funkcija, dok se dodatne funkcionalnosti implementiraju kroz nezavisne module ili dodatke (eng. *plug-in*).

Svi moduli komuniciraju sa jezgrom sistema, koje posreduje u njihovoj međusobnoj koordinaciji. Ovakav pristup omogućava visok stepen proširivosti i fleksibilnosti, jer se nove funkcionalnosti mogu dodavati ili menjati bez potrebe za modifikacijom osnovne strukture sistema. Zbog toga se mikrokernel arhitektura često primenjuje u sistemima koji zahtevaju visok stepen modularnosti, prilagodljivosti i dugoročne održivosti, kao što su operativni sistemi.



Slika 2.4: Mikrokernel arhitektura sa centralnim jezgrom i proširivim modulima.

Ukoliko se sve komponente sistema izvršavaju na jednom računarskom uređaju, sistem zadržava **centralizovanu** prirodu. Međutim, kada se pojedini moduli izvršavaju na različitim fizičkim uređajima koji međusobno koordiniraju svoje stanje, sistem prelazi u kategoriju **distribuiranih sistema**.

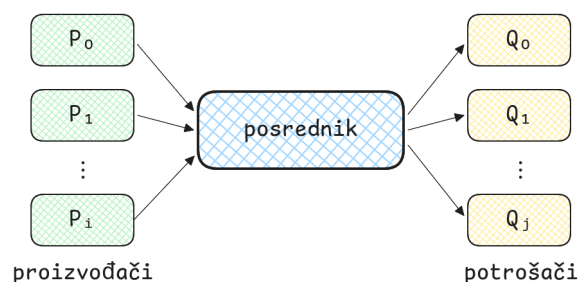
Definicija 2.11. *Distribuiran informacioni sistem predstavlja skup nezavisnih računarskih uređaja (čvorova) koji ne dele zajedničku memoriju i međusobno komuniciraju razmenom poruka, pri čemu koordinacijom svog rada prema korisniku deluju kao jedinstven i koherentan sistem [26].*

Za razliku od centralizovanih sistema, u distribuiranim sistemima komunikacija između komponenti ne može se ostvariti direktnim pozivima funkcija, već se realizuje putem mrežnih komunikacionih mehanizama. Najčešći obrasci komunikacije uključuju:

1. Interfejs za programiranje aplikacija [27] (eng. *Application Programming Interface, API*) - skup definisanih pravila i interfejsa koji omogućavaju komunikaciju između aplikacija preko mreže,
2. Udaljeni poziv procedure [28] (eng. *Remote Procedure Call, RPC*) - mehanizam koji omogućava pozivanje funkcija na udaljenim sistemima kao da su lokalne,
3. Red poruka [29] (eng. *Message Queue*) - asinhroni mehanizam razmene poruka putem posredničkog sistema za razmenu poruka,
4. Tok događaja [29] (eng. *Event Streaming*) - mehanizam kontinuirane razmene i obrade tokova događaja u realnom vremenu,
5. Web komunikaciona veza [30] (eng. *Web Socket*) - mehanizam koji omogućava stalno otvorenu, dvosmernu komunikaciju između klijenta i servera preko jedne konekcije.

Arhitektura vođena događajima [31]. Arhitektura vođena događajima zasniva se na razmeni informacija putem događaja koji nastaju u sistemu. Komponente koje generišu događaje nazivaju se proizvođači (eng. *producers*), dok komponente koje ih obrađuju predstavljaju potrošači (eng. *consumers*). Komunikacija između njih često se ostvaruje posredstvom sistema za posredovanje poruka (eng. *broker*).

Najčešće implementacije ovog pristupa zasnivaju se na sistemima redova poruka ili sistemima za tok događaja. Kod sistema redova poruka svaku poruku obrađuje jedan potrošač, dok kod sistema za tok događaja više potrošača može istovremeno primiti i obraditi isti događaj.



Slika 2.5: Arhitektura vođena događajima sa proizvođačima, posrednikom i potrošačima događaja.

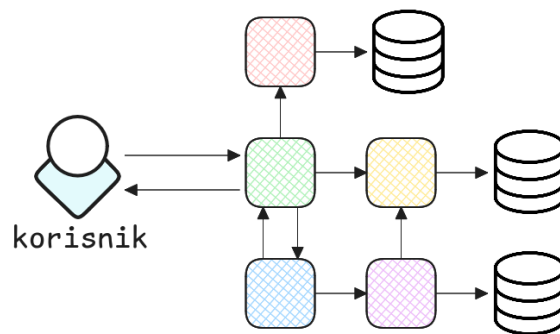
Ovakav pristup povećava fleksibilnost sistema jer omogućava jednostavno dodavanje novih proizvođača i potrošača. Međutim, posrednik može postati usko grlo sistema, zbog čega se u

praksi često koristi klaster (eng. *cluster*) posrednika koji raspodeljuje opterećenje između više instanci.

Mikroservisna arhitektura [32]. Mikroservisna arhitektura zasniva se na principu da svaka komponenta sistema ima jednu jasno definisanu odgovornost (eng. *single responsibility*). Sistem je sastavljen od većeg broja malih, samostalnih servisa koji međusobno komuniciraju putem mreže, najčešće korišćenjem *API*-ja ili *RPC* mehanizama.

Svaki mikroservis obično predstavlja kompletan mini-sistem sa sopstvenim ulaznim i izlaznim interfejsima, poslovnom logikom i često sopstvenim skladištem podataka. Zbog takve strukture mikroservisi se mogu nezavisno razvijati, testirati, skalirati i ažurirati.

Ovaj pristup povećava fleksibilnost i skalabilnost sistema, kao i mogućnost integracije sa drugim aplikacijama. Međutim, istovremeno uvodi dodatnu složenost u pogledu orkestracije servisa, bezbednosti komunikacije i nadgledanja distribuiranih komponenti.



Slika 2.6: Mikroservisna arhitektura sa nezavisnim servisima i mrežnom komunikacijom.

Navedene arhitekture predstavljaju najrasprostranjenije pristupe organizaciji savremenih informacionih sistema, iako u praksi postoje i drugi specijalizovani arhitektonski modeli namenjeni specifičnim domenima primene.

Dodatno, navedene arhitekture koje se izvršavaju na jednom uređaju moguće je proširiti pokretanjem više instanci i uvođenjem raspoređivanja opterećenja, pri čemu se prema potrebi bira odgovarajući komunikacioni mehanizam između njih.

2.4.1 Računarstvo u oblaku i elastično skaliranje informacionih sistema

Razvoj informacionih sistema tradicionalno je zahtevao nabavku sopstvene fizičke infrastrukture (servera, mrežne opreme, skladišta podataka) i angažovanje stručnog osoblja za njeno održavanje. Takav pristup nosi visoke troškove održavanja infrastrukture, zahteva složeno planiranje i otežava brzo prilagođavanje promenama u obimu posla.

Savremene organizacije sve češće usvajaju model računarstva u oblaku (eng. *cloud computing*) [33], u kome se infrastruktura, platforme i/ili softverske usluge iznajmljuju od eksternog provajdera.

Umesto sopstvene opreme koriste se virtuelni resursi koji se dinamički dodeljuju prema potrebi, čime se omogućava elastično skaliranje instanci informacionog sistema, smanjuju troškovi infrastrukture i održavanja, umanjuje potreba za internim sistemskim osobljem i ubrzava postavljanje razvojnih i produkcionih okruženja.

Okruženja u oblaku tipično se realizuju kao javna, privatna ili hibridna, dok se isporuka servisa organizuje kroz standardne modele: *IaaS* (infrastruktura kao usluga), *PaaS* (platforma kao usluga) i *SaaS* (softver kao usluga). Ovi modeli određuju nivo kontrole koji zadržava korisnik i nivo odgovornosti koji preuzima provajder usluge.

Kontejnerizacija predstavlja dominantan tehnološki pristup za raspoređivanje softverskih komponenti u oblaku. Kontejneri (npr. *Docker*) omogućavaju izolovano, prenosivo i konzistentno pokretanje aplikacionih instanci, dok platforme za orkestraciju (npr. *Kubernetes*) automatizuju njihovo raspoređivanje i oporavak u dinamičkim okruženjima.

Zahvaljujući ovim tehnologijama, arhitektonske organizacije informacionih sistema mogu se efikasno distribuirati i održavati u *cloud* infrastrukturi, uz značajno smanjenje tehničke složenosti i operativnih troškova u odnosu na tradicionalne modele upravljanja resursima.

2.5 Centralizacija u distribuiranim informacionim sistemima

Savremeni informacioni sistemi često se izvršavaju na velikom broju fizički distribuiranih čvorova, mada njihova dominantna arhitektonska paradigma mrežne komunikacije i dalje ostaje topološki klijent-server model [34]. U takvom modelu unapred je definisan raspored komunikacije između komponenti sistema, pri čemu pojedini čvorovi imaju ulogu servera koji pružaju usluge, dok drugi čvorovi nastupaju kao klijenti koji te usluge zahtevaju. Posledično, komunikaciona struktura sistema ima hijerarhijski karakter, jer čvorovi ne poseduju ravnopravan položaj u mreži.

Iako je infrastrukturno raspodeljen na više računarskih čvorova, ovakav sistem zadržava centralizovan model upravljanja. Poslovna logika, donošenje ključnih odluka i upravljanje stanjem sistema najčešće su koncentrisani u okviru jedinstvene upravljačke instance ili ograničenog broja međusobno koordinisanih autoriteta. Zbog toga se sistem, sa operativne tačke gledišta, ponaša kao jedinstvena računarska celina sa jednim globalnim stanjem i jednim centrom poverenja.

Međutim, istorijski razvoj dominantnih modela digitalnih platformi bio je pre svega usmeren na optimizaciju poslovnih ciljeva i tržišne vrednosti organizacija koje njima upravljaju. Kao posledica takvog pristupa, kontrola nad podacima, pravilima interakcije i distribucijom vrednosti u velikoj meri ostaje koncentrisana u rukama centralnih upravljačkih entiteta.

U savremenoj literaturi sve češće se ukazuje na to da centralizovana kontrola nad informacionim sistemima može dovesti do smanjenja autonomije korisnika i pojave različitih oblika asimetrije moći između platformi i njihovih korisnika. Među najčešće diskutovanim pojavama izdvajaju se:

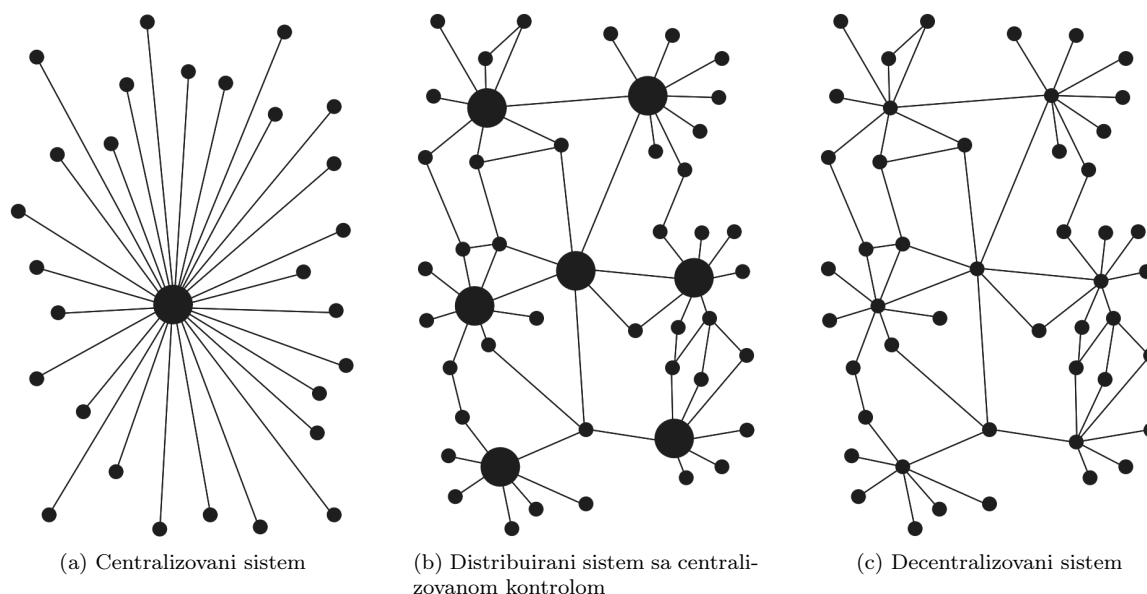
1. **Algoritamska kontrola** [35] - korišćenje algoritama za filtriranje, rangiranje i preporuku sadržaja, čime platforme indirektno utiču na informacione tokove kojima su korisnici izloženi,
2. **Cenzura i kontrola pristupa** [36] - centralizovano upravljanje sadržajem i pristupom platformama, koje omogućava uklanjanje sadržaja ili ograničavanje pristupa određenim korisnicima,
3. **Monopol i koncentracija moći** [37] - koncentracija tržišne moći u okviru velikih digitalnih platformi može ograničiti konkurenciju i uticati na izbor proizvoda i usluga dostupnih korisnicima.
4. **Nadzorni kapitalizam** [38] - ekonomski model digitalnih platformi zasnovan na masovnom prikupljanju i analizi korisničkih podataka radi optimizacije poslovnih modela i ciljanog oglašavanja,

5. **Zloupotreba ličnih podataka** [39] - koncentracija velikih količina korisničkih podataka u centralizovanim bazama povećava rizike od neovlašćenog pristupa, curenja podataka i drugih oblika narušavanja privatnosti,

Da bi se informacioni sistem mogao smatrati decentralizovanim, nije dovoljno da se izvršava na velikom broju čvorova. Neophodno je da njegova arhitektura bude projektovana bez centralne tačke kontrole ili jedinstvene tačke otkaza sistema (eng. *single point of failure*), jer bi njen otkaz, uklanjanje ili kompromitovanje dovelo do poremećaja ili potpunog prekida rada sistema [40].

Prema standardnoj klasifikaciji [41], decentralizacija se može analizirati kroz tri međusobno nezavisne dimenzije:

1. **Arhitektonska decentralizacija** - fizička i geografska raspodela računarskih čvorova,
2. **Kontrolna (politička) decentralizacija** - raspodela moći odlučivanja i upravljanja među različitim učesnicima,
3. **Logička decentralizacija** - struktura softverskog stanja i način na koji sistem konceptualno funkcioniše kao celina.



Slika 2.7: Arhitektonski modeli organizacije i upravljanja informacionim sistemima.

2.5.1 Arhitektonska decentralizacija

Arhitektonska decentralizacija odnosi se pre svega na **hardverski sloj** sistema i ne predstavlja binarnu osobinu sistema, već kontinualni spektar koji zavisi od broja čvorova, njihove međusobne nezavisnosti, vlasničke strukture i fizičke raspodele.

Definicija 2.12. *Arhitektonska decentralizacija podrazumeva postojanje fizičke infrastrukture sastavljene od čvorova koje pokreću međusobno nezavisni vlasnici, raspoređenih na različitim fizičkim lokacijama, bez centralizovanog mehanizama upravljanja infrastrukturom.*

Tipičan primer sistema u kome su prisutne sve tri forme centralizacije jesu tradicionalne korporativne infrastrukture, gde se sistem izvršava na više fizički i geografski razdvojenih lokacija, ali su svi ti resursi u vlasništvu ili pod kontrolom jednog administrativnog entiteta i podležu jedinstvenoj upravljačkoj politici.

Posledica. *Fizička distribuiranost čvorova ne implicira decentralizaciju informacionog sistema.*

2.5.2 Kontrolna decentralizacija

Kontrolna (politička) decentralizacija odnosi se na **ljudski i organizacioni sloj** sistema i podrazumeva odsustvo jedinstvenog entiteta koji poseduje ovlašćenje da izdaje naredbe, donosi konačne odluke ili menja globalno stanje sistema.

Definicija 2.13. *Kontrolna decentralizacija označava svojstvo sistema u kojem nijedan pojedinačni čvor ili učesnik nema privilegovanu ulogu u donošenju odluka, već se napredovanje sistema ostvaruje kolektivnim delovanjem nezavisnih učesnika, bez postojanja centralizovanog autoriteta ili tačke poverenja.*

Da bi ovakav model bio moguć, čvorovi sistema moraju biti u mogućnosti da međusobno komuniciraju direktno, bez posrednika sa centralnom ulogom koji bi mogli da nameću pravila komunikacije.

Definicija 2.14. *Mreža ravnopravnih korisnika (eng. peer-to-peer, P2P) predstavlja model komunikacije u kome svaki učesnik može preuzimati ulogu i klijenta i servera, omogućavajući direktan prenos podataka između dva ili više čvorova bez oslanjanja na centralni server [42].*

Iako ovakva topologija omogućava eliminisanje *centralne tačke komunikacije* i predstavlja osnovu arhitektonske decentralizacije, ona sama po sebi ne garantuje ostvarivanje kontrolne decentralizacije. Naime, ona ne eliminiše centralizovane mehanizme kontrole nad pravilima sistema, kao što su pravo učešća u mreži, validnost poruka i način donošenja odluka o napredovanju globalnog stanja.

Teorema 2.1. *Mreža ravnopravnih korisnika predstavlja potreban, ali ne i dovoljan uslov za ostvarivanje kontrolne decentralizacije.*

Iz tog razloga, sam sistem mora formalno definisati procese i aktivnosti čijim se izvršavanjem omogućava da svi čvorovi, na osnovu lokalnih informacija i razmene poruka, dinamički odrede napredovanje sistema i kolektivno ga prihvate.

Međutim, uklanjanje centralizovanog autoriteta u procesima odlučivanja implicitno uvodi mogućnost da se pojedini čvorovi ponašaju neispravno ili maliciozno, bilo usled tehničkih kvarova, bilo kao posledica kompromitovanja od strane napadača koji šalju lažne ili međusobno kontradiktorne poruke.

Ovakvi incidenti ilustruju izazov poznat kao *Problem vizantijskih generala* [43] (eng. *Byzantine Generals Problem*), u kojem sistem mora da obezbedi konzistentan dogovor čak i u prisustvu čvorova koji se ponašaju maliciozno ili nepredvidivo.

Definicija 2.15. *Vizantijski general je učesnik u distribuiranom sistemu koji je uključen u koordinaciju ili donošenje zajedničke odluke, ali može biti zlonamern, neispravan ili namerno šalje netačne ili kontradiktorne informacije drugim učesnicima.*

U distribuiranim sistemima, mogućnost da se pojedini učesnici ponašaju vizantijski predstavlja jedan od fundamentalni izazova za očuvanje korektnog i konzistentnog napredovanja sistema. Ukoliko vizantijski čvorovi raspolazu većinskim udelom resursa koji su relevantni za donošenje odluka

i održavanje globalnog stanja sistema, bilo da se radi o broju čvorova, računarskoj snazi, udelu uloga ili drugim oblicima sistemskog uticaja, oni mogu preuzeti efektivnu kontrolu nad procesom odlučivanja, čime se narušavaju sigurnost, konzistentnost i kontrolna decentralizacija sistema.

Iz tog razloga, svaki distribuirani sistem mora eksplicitno definisati pretpostavke o maksimalnom obimu vizantijskog ponašanja koji je u stanju da toleriše, što predstavlja njegovo fundamentalno teorijsko ograničenje.

Teorema 2.2. *Ako se u mreži ravnopravnih korisnika ostvaruje kontrolna decentralizacija, sistem mora biti projektovan tako da definiše gornju granicu udela sistemskih resursa koji mogu biti pod kontrolom vizantijskih učesnika, unutar koje se garantuje korektno funkcionisanje sistema.*

U slučaju prekoračenja ove granice, vizantijski učesnici mogu manipulacijom porukama, pravilima validacije ili procesom donošenja odluka proizvesti divergentna stanja sistema, onemogućiti postizanje konsenzusa ili nametnuti sopstvenu verziju globalnog stanja, čime se efektivno urušava kontrolna decentralizacija.

Postoji veliki broj algoritama za rešavanje problema vizantijskih generala [43]. Međutim, pokazano je da je za postizanje determinističkog i sigurnog konsenzusa u prisustvu do f vizantijskih čvorova neophodno da ukupan broj učesnika zadovoljava uslov $n \geq 3f + 1$. Ovaj uslov predstavlja donju granicu na broj čvorova potrebnih za tolerisanje f proizvoljno neispravnih učesnika, čime se obezbeđuje da ispravni čvorovi čine strogu većinu u odnosu na potencijalno zlonamerne.

2.5.3 Logička decentralizacija

Logička decentralizacija odnosi se na **softverski i protokolarni sloj** sistema i opisuje način na koji je sistem konceptualno organizovan, odnosno kako se njegovo stanje modeluje, deli i koristi od strane učesnika.

Za razliku od arhitektonske i kontrolne decentralizacije, koje se bave fizičkom infrastrukturom i raspodelom ovlašćenja, logička decentralizacija se odnosi na *apstraktni oblik sistema*, odnosno na to da li se on posmatra kao jedinstvena, nedeljiva celina ili kao skup lokalno povezanih, samostalnih delova.

Definicija 2.16. *Logička decentralizacija označava konceptualnu organizaciju sistema u kojoj ne postoji jedinstveno, centralno definisano globalno stanje koje svi čvorovi moraju da dele i održavaju, već se sistem sastoji od više lokalnih stanja koja su međusobno nezavisna i čija se ispravnost ne oslanja na postojanje ili dostupnost ostatka sistema.*

Intuitivno, sistem je logički decentralizovan ukoliko se može podeliti na dva ili više delova tako da svaki deo, zajedno sa svojim korisnicima i resursima, i dalje funkcioniše na način koji je u skladu sa originalnom namenom sistema, bez potrebe za koordinacijom sa ostalim delovima.

Primer sistema koji je logički decentralizovan, ali kontrolno centralizovan, predstavlja mreža za isporuku sadržaja [44] (eng. *Content Delivery Network, CDN*). Ona se sastoji od velikog broja geografski raspoređenih rubnih servera (eng. *edge servers*), koji služe za efikasnu isporuku statičkog sadržaja krajnjim korisnicima. Istu infrastrukturu može koristiti više nezavisnih proizvođača sadržaja, pri čemu svaki rubni server održava sopstvene lokalne kopije podataka. U takvom sistemu ne postoji jedinstveno operativno globalno stanje koje svi serveri moraju u svakom trenutku da dele. Ukoliko se infrastruktura podeli na dva odvojena dela, svaki deo i dalje može samostalno isporučivati sadržaj svojim korisnicima. Iako se izmene sadržaja iniciraju iz centralnog izvora, sama isporuka i

osnovno funkcionisanje sistema ne zavise od postojanja jedinstvene, centralno usklađene slike stanja u celokupnoj mreži.

Primer sistema koji ostvaruje i logičku i kontrolnu decentralizaciju predstavlja *BitTorrent* protokol [45]. BitTorrent funkcioniše kao mreža ravnopravnih korisnika u kojoj se za svaki sadržaj formira posebna grupa čvorova koji učestvuju u njenoj razmeni. Svaki čvor održava isključivo sopstveno lokalno stanje, koje se odnosi na delove sadržaja koje poseduje i razmenjuje. U ovom sistemu ne postoji globalno stanje koje bi objedinjavalo sve učesnike mreže. Različite grupe čvorova funkcionišu nezavisno jedna od druge, a podela sistema na više delova ne narušava njegovu osnovnu funkcionalnost. Svaki deo mreže i dalje omogućava pronalaženje, razmenu i preuzimanje sadržaja u okviru svojih lokalnih interakcija. Servisi za pretragu sadržaja (npr. *The Pirate Bay*) mogu olakšati pronalaženje relevantnih grupa korisnika, ali nisu neophodni za osnovno funkcionisanje sistema i ne predstavljaju njegovu logičku komponentu.

2.6 Granice i implikacije decentralizacije u distribuiranim informacionim sistemima

Od tri dimenzije decentralizacije, kontrolna decentralizacija u praksi obično ima najveću intuitivnu težinu. Kada se za neki sistem kaže da je „decentralizovan”, najčešće se pod tim podrazumeva da korisnici nisu podređeni jednom centralnom autoritetu i da nijedan pojedinačni subjekt nema potpunu kontrolu nad upravljanjem sistemom.

Zbog toga je moguće da sistem bude arhitektonski ili logički centralizovan, a da se i dalje smatra decentralizovanim ukoliko je kontrola nad njegovim funkcionisanjem raspodeljena između većeg broja učesnika. Međutim, arhitektonska centralizacija uvodi jedinstvenu tačku otkaza sistema, što predstavlja značajan rizik za pouzdanost i otpornost informacionih sistema.

U praksi, većina savremenih distribuiranih informacionih sistema zadržava kontrolno centralizovan model upravljanja, dok se distribucija infrastrukture koristi prvenstveno radi povećanja dostupnosti, skalabilnosti i performansi sistema. Takvi sistemi uvode dodatnu složenost u odnosu na centralizovane arhitekture, jer zahtevaju mehanizme sinhronizacije stanja između čvorova, upravljanje konkurentnim izvršavanjem i rešavanje problema konzistentnosti podataka. Ovi mehanizmi neminovno utiču na performanse sistema, a samim tim i na korisničko iskustvo.

Uvođenje dodatnih oblika decentralizacije u takvim okruženjima dodatno povećava složenost sistema, jer zahteva razvoj novih mehanizama koordinacije koji mogu da obezbede pouzdano upravljanje zajedničkim stanjem sistema bez centralnog autoriteta.

Iz tog razloga decentralizacija se može posmatrati kao proširenje sociotehničke perspektive informacionih sistema, primenjeno na domene distribuiranih algoritama, konsenzusa i upravljanja zajedničkim stanjem u mrežama nezavisnih učesnika.

Analiza takvih sistema zahteva oslanjanje na teorijske osnove distribuiranih sistema, koje proučavaju ograničenja i kompromise između skalabilnosti, konzistentnosti, bezbednosti i otpornosti na otkaze u okruženjima bez centralne tačke kontrole.

U narednom poglavlju biće predstavljeni osnovni koncepti i metrike teorije distribuiranih sistema koji omogućavaju formalnu analizu ovih ograničenja. Ovaj teorijski okvir poslužiće kao osnova za razumevanje blokčejn tehnologija, koje u kasnijim poglavljima posmatramo kao posebnu klasu distribuiranih informacionih sistema usmerenih ka uklanjanju centralnog autoriteta nad upravljanjem stanjem sistema.

Glava 3

Analitički okvir za evaluaciju informativnih sistema

Cilj ovog poglavlja jeste da uspostavi analitički okvir za analizu tehničkih prednosti i ograničenja informativnih sistema, nezavisno od toga da li su oni centralizovani, distribuirani ili decentralizovani.

Da bi se obezbedio standardizovan okvir evaluacije, karakteristike informativnih i softverskih arhitektura formalizuju se kroz **atribute kvaliteta** prema IEEE i ISO standardima [46, 47]. Za potrebe ove disertacije atributi kvaliteta biće objedinjeni i grupisani u sledeće analitičke kategorije:

1. **Performanse i efikasnost:** performanse, skalabilnost, efikasnost korišćenja resursa, troškovna efikasnost, operativno održivost, fleksibilnost i energetska efikasnost.
2. **Operativna svojstva:** dostupnost, integritet, sigurnost, pouzdanost, otpornost na greške, konzistentnost podataka.
3. **Upotrebljivost:** korisničko iskustvo, lakoća učenja (eng. *learnability*), operabilnost, pristupačnost.
4. **Funkcionalna ispravnost:** ispravnost funkcionalnih zahteva, potpunost funkcija, preciznost izvođenja funkcija, usklađenost sa specifikacijom.

Fokus je isključivo na **modelu resursa informacione tehnologije**, koji obuhvata infrastrukturu i tehnološke resurse čije korišćenje i održavanje omogućavaju realizaciju i podršku sistema. Menadžerski, organizacioni i ekonomski faktori pominju se samo okvirno, jer ne predstavljaju predmet detaljne analize ove disertacije.

Ovakva klasifikacija omogućava konzistentno i teorijski utemeljeno poređenje informativnih sistema na tehničkom nivou. Razmatranje ovih atributa omogućava jasno razdvajanje tehničkih svojstava sistema, koja predstavljaju primarni predmet istraživanja u ovoj disertaciji, od šireg organizacionog i poslovnog konteksta.

3.1 CAP teorema

Jedno od najznačajnijih kvalitativnih ograničenja u distribuiranim sistemima proizilazi iz činjenice da su **mrežne particije** neizbežan događaj: pitanje nije „da li” će se dogoditi, već „kada”. Ova

činjenica je najjasnije formalizovana kroz *CAP teoremu*, koja definiše osnovni skup arhitektonskih kompromisa u dizajnu distribuiranih sistema.

Mrežna particija može nastati u različitim situacijama, uključujući:

1. *prekid mrežne povezanosti* - fizički prekid veze između čvorova koji nastavljaju da rade, ali više ne mogu da komuniciraju,
2. *ekstremno povećanje vremena odaziva* - čvor je operativan ali vreme odaziva datog čvora prelazi prag koji je prihvatljiv (npr. skok sa 50 ms na nekoliko minuta), pa ga ostatak sistema tretira kao nedostupan,
3. *greške u mrežnoj konfiguraciji* - ispravni čvorovi ne vide jedni druge zbog pogrešne konfiguracije.

Ponašanje sistema u ovakvim uslovima je prvi formulisao Eric Brewer [48], a formalno su dokazali Gilbert i Lynch [49]. Teorema definiše osnovni kompromis sa kojim se suočavaju svi distribuirani sistemi koji upravljaju deljenim stanjem.

Teorema 3.1. *U distribuiranom sistemu koji čuva podatke moguće je istovremeno garantovati najviše dve od sledeće tri osobine:*

- (1) **Konzistentnost** (*eng. Consistency, C*) - svaki čvor vidi isto stanje sistema, tj. operacija čitanja uvek vraća poslednju potvrđenu upisanu vrednost,
- (2) **Dostupnost** (*eng. Availability, A*) - sistem garantuje odgovor na svaki zahtev u konačnom vremenu,
- (3) **Toleranciju na particije** (*eng. Partition Tolerance, P*) - sistem nastavlja da radi i u slučaju mrežnih particija (poruke se mogu izgubiti ili stići sa proizvoljnim kašnjenjem).

Posledica. *Kada dođe do mrežne particije, distribuiran sistem ne može istovremeno garantovati punu konzistentnost i punu dostupnost. Stoga je prinuđen da da prioritet ili **konzistentnosti** (CP sistemi) ili **dostupnosti** (AP sistemi).*

3.2 Performanse i efikasnost

Radni sistem ostvaruje interakciju sa informacionim sistemom kroz zahteve koji predstavljaju formalizovani izraz poslovnih potreba. Obradom pristiglih zahteva informacioni sistem izvršava odgovarajuće operacije nad svojim stanjem, čime se ostvaruje njegovo napredovanje i realizuju tražene funkcionalnosti.

Definicija 3.1. *Zahtev predstavlja formalizovanu specifikaciju funkcionalnosti ili promene stanja koju informacioni sistem treba da realizuje. Finalizacija zahteva označava trenutak kada informacioni sistem završi kompletnu obradu zahteva.*

Zahtev može da inicira:

1. krajnji korisnik,
2. operater,
3. drugi informacioni sistem,
4. servis unutar istog sistema,

5. automatizovani proces.

Tipični primeri zahteva uključuju upis podatka u bazu, izvršavanje funkcionalnosti sistema, dohvrat ili transformaciju informacija, i transakcije koje menjaju stanje sistema. Finalizacija zahteva može da podrazumeva vraćanje rezultata inicijatoru ili trajno beleženje izvršene operacije.

U skladu sa savremenim pristupima modeliranju procesa i *end-to-end* analizi radnih tokova [50], **zahtev se ne posmatra kao izolovana jedinica**, već kao deo šireg procesa čija se vrednost i funkcionalna uloga ispoljava tek u interakciji sa ostalim procesima i komponentama arhitekture informacionog sistema.

Iz perspektive korisnika ili radnog sistema relevantan je rezultat obrade zahteva, dok se iz perspektive informacionog sistema posmatraju resursi, vreme i operacije neophodne za njegovu realizaciju.

Definicija 3.2. *Performanse predstavljaju skup kvantitativnih pokazatelja koji opisuju efikasnost obrade zahteva u informacionom sistemu.*

Kako su performanse apstraktan pojam koji se kvantitativno izražava različitim metrikama, a pri tom različiti učesnici mogu imati različite kriterijume „efikasnost” sistema (npr. organizacije prvenstveno optimizuju troškove, dok krajnji korisnici vrednuju brzinu i cenu usluge), u daljem radu razmatraju se sledeće ključne performansne metrike:

1. **vreme odaziva (eng. *latency*)** - vreme koje protekne od slanja zahteva do dobijanja odgovora,
2. **propusnost (eng. *throughput*)** - broj zahteva koje sistem može da obradi u jedinici vremena,
3. **cena obrade (eng. *cost*)** - trošak izvršavanja pojedinačnog zahteva.

Zajedničko za sve navedene metrike jeste da na njih direktno utiču tri fundamentalna resursa informacionog sistema: hardver (procesor, memorija, I/O podsistem), softver (algoritmi, implementacija, organizacija baze podataka) i mreža (latencija, širina opsega, pouzdanost i stabilnost veze).

Definicija 3.3. *Kapacitet predstavlja maksimalnu dugoročno održivu stopu obrade zahteva koju informacioni sistem može da postigne u stabilnom režimu rada, bez degradacije definisanih performansnih metrika [51].*

Međutim, kada hardver, softver i/ili mrežni sloj ne mogu da obrade broj pristiglih zahteva, informacioni sistem izlazi iz stabilnog režima rada i ulazi u stanje zagušenja.

Definicija 3.4. *Zagušenje predstavlja stanje informacionog sistema u kojem je stopa pristiglih zahteva veća od kapaciteta.*

U stanju zagušenja, zahtevi se akumuliraju brže nego što ih sistem može obraditi, redovi čekanja rastu, latencija postaje nepredvidiva, a pojedine operacije postaju neizvodljive u datim vremenskim ograničenjima.

Primena fundamentalnih rezultata teorije čekanja [52] omogućava formalno razumevanje ponašanja sistema kada dolazna stopa zahteva premaši njegov kapacitet, što se direktno preslikava na posmatrane performansne metrike. Time dobijamo:

Teorema 3.2. *Kada dolazna stopa zahteva premaši kapacitet sistema, vreme odaziva raste bez ograničenja, propusnost dostiže svoj maksimum, a cena obrade svakog zahteva eksponencijalno raste usled akumulacije redova čekanja i pojačane konkurencije za resurse.*

U stvarnim informacionim sistemima obim pristiglih zahteva može u određenim trenucima da premaši raspoloživi kapacitet sistema. Zato je od suštinskog značaja razumeti kako se kapacitet može povećavati dodavanjem dodatnih resursa (računske snage, memorije, mrežnog kapaciteta, dodatnih čvorova itd.). U tom kontekstu definiše se sledeće svojstvo:

Definicija 3.5. *Skalabilnost predstavlja sposobnost informacionog sistema da se uz povećanje dostupnih resursa ostvari porast kapaciteta obrade zahteva, uz očuvanje definisanih performansnih karakteristika [53].*

Definicija 3.6. *Optimizacija predstavlja proces unapređenja softverskog sistema sa ciljem da za isti funkcionalni zahtev ostvari bolje performanse, najčešće kroz smanjenje upotrebe tehničkih resursa poput vremena izvršavanja, memorije, mrežnog protoka ili energije, pri čemu se zadržava nepromenjeno spoljašnje ponašanje sistema.*

Skaliranje se odnosi na način proširenja hardverskih resursa i optimizacije softverskih aplikacija, te uključuje i hardverske i softverske aspekte sistema. U praksi se, nakon sprovedene softverske optimizacije, razlikuju dva osnovna pristupa skaliranju hardvera:

1. **Vertikalno skaliranje (*scale-up*)** - povećavanje resursnih kapaciteta jedne instance sistema (CPU, memorija, skladište),
2. **Horizontalno skaliranje (*scale-out*)** - dodavanje novih instanci (ili procesa) unutar istog informacionog sistema, čime se povećava ukupan broj aktivnih izvršnih jedinica.

U praksi se koristi i *dijagonalno skaliranje* koje predstavlja kombinovanje vertikalnog i horizontalnog pristupa.

3.2.1 Vertikalno skaliranje

Vertikalno skaliranje podrazumeva povećavanje resursnih kapaciteta, tj. procesorskih resursa, memorije, I/O podsistema ili skladišnog prostora jedne fizičke ili virtuelne instance informacionog sistema. Time se jedna izvršna jedinica osposobljava da obradi veći broj zahteva u jedinici vremena. Ovakvo skaliranje je idealno za jednostavnije sisteme, gde bi se koristila monolitna, slojevita ili mikrokernel arhitektura.

Prednosti vertikalnog skaliranja obuhvataju:

1. **Jednostavnija arhitektura** - kompletan informacioni sistem izvršava se na jednoj fizičkoj ili virtuelnoj instanci, što smanjuje arhitektonsku složenost,
2. **Konzistentnost** - sve operacije čitanja i pisanja odvijaju se lokalno čime se obezbeđuje stabilna i deterministička konzistentnost,
3. **Niska operativna složenost** - sistem je jednostavniji za praćenje, održavanje, otklanjanje grešaka i nadgledanje performansi,
4. **Minimalne promene u softveru** - aplikacija najčešće može da koristi jače resurse bez potrebe za promenom arhitekture, algoritama ili načina obrade podataka.

Međutim, postoje ozbiljna ograničenja:

1. **Hardverski limit** - svaka mašina ima gornju granicu kapaciteta: maksimalan broj CPU jezgara, količinu RAM-a, maksimalnu brzinu I/O-a i skladišta. Jednom kada se taj limit dostigne, više nije moguće dodatno skalirati,
2. **Ekonomska neefikasnost na visokom nivou** - cena raste brzo i nelinearno u odnosu na dobijene performanse. Na primer, povećanje broja jezgara ne donosi linearnu dobit performansi [54],
3. **Opterećenje** - kada sistem dođe do zagušenja i resursi budu iscrpljeni, performanse mogu naglo da opadnu, uz značajan porast latencije, kašnjenja ili čak potpuni pad servisa. Kada se dostigne hardverski limit, ne postoji način daljeg skaliranja bez arhitekturne promene, a period oporavka može biti dug,
4. **Jedna tačka otkaza** - dostupnost celokupnog sistema direktno zavisi od ispravnosti jedne mašine, njen pad ili otkaz dovodi do potpunog prekida rada sistema.

Da bi se makar delimično ublažile poslednje dve mane, jedno od poznatih rešenja jeste korišćenje rezerve (eng. *hot standby*). To znači da postoji rezervna instanca spremna za aktivaciju u slučaju da glavna mašina zakaže ili postane neupotrebljiva. Ovo omogućava preuzimanje rada bez (ili uz minimalan) prekid za korisnike. Ipak, i ta rezerva nasleđuje sve prethodno navedene mane.

3.2.2 Lokacijski zasnovano adresiranje

Većina informacionih sistema omogućava komunikaciju između geografski raspoređenih klijenata i instance informacionog sistema, budući da se ta komunikacija zasniva na modelu **lokacijski zasnovanog adresiranja** (eng. *location-based addressing*) [55]. U ovom modelu svaki zahtev se eksplicitno upućuje na unapred poznatu lokaciju servera, pri čemu se koristi DNS servis koji prevodi naziv domena u odgovarajuću IP adresu. Na taj način se celokupan radni tok obrade zahteva usmerava ka jednoj fizičkoj lokaciji.

Ovakav model adresiranja jednostavan je za implementaciju i predstavlja dominantnu paradigmu tradicionalnih internet servisa. Međutim, on ima ključna ograničenja u pogledu performansi kada se klijenti i server nalaze na različitim geografskim lokacijama. Brojna merenja vremena odaziva globalnog interneta pokazuju da pristup serverima lociranim na drugim kontinentima tipično uvodi vreme odaziva od **80-150 ms** [56] po jednom zahtevu, što može značajno uticati na *korisničko iskustvo* kod aplikacija osetljivih na kašnjenje, kao što su interaktivne web aplikacije, sistemi za finansijsko trgovanje, igre u realnom vremenu i *streaming* servisi.

Usled ovih ograničenja, vertikalno skalirani centralizovani sistemi ne mogu da kompenzuju geografska kašnjenja, nezavisno od raspoloživog hardvera. Ovaj problem se ublažava primenom geografski distribuiranih instanci, pri čemu se zahtevi usmeravaju ka najbližem dostupnom čvoru, čime se smanjuje ukupno vreme odaziva.

3.2.3 Horizontalno skaliranje

Horizontalno skaliranje podrazumeva povećanje kapaciteta informacionog sistema dodavanjem dodatnih instanci aplikacije (procesa, kontejnera ili servera), pri čemu se svaki novi zahtev može obraditi na bilo kojoj od dostupnih instanci. Na taj način ukupan kapacitet sistema raste, budući da se opterećenje raspodeljuje na veći broj izvršnih jedinica.

Prednosti horizontalnog skaliranja obično se klasifikuju na sledeći način:

1. **Teoretski neogračen kapacitet** - budući da se kapacitet može povećavati dodavanjem novih instanci, sistem nije vezan za jedinstveno hardversko ograničenje,
2. **Ekonomska efikasnost** - troškovi skaliranja rastu približno linearno u odnosu na povećanje kapaciteta, što omogućava postepeno, predvidivo i finansijski održivo proširenje,
3. **Geografska distribucija** - instance se mogu rasporediti na različite fizičke lokacije, bliže krajnjim korisnicima, čime se smanjuje vreme odaziva i ublažava problem pristupa udaljenim serverima kod lokacijski zavisnog adresiranja,
4. **Otpornost na greške** - izostanak jedne tačke otkaza omogućava da kvar pojedinačne instance ne utiče na dostupnost celog sistema.

Uočavamo da se mnoge slabosti vertikalnog skaliranja u horizontalnom pristupu transformišu u njegove prednosti. Međutim, istovremeno se prednosti vertikalnog pristupa pretvaraju u izazove svojstvene distribuiranim sistemima. Ovo je posledica činjenice da održavanje većeg broja instanci nužno zahteva uvođenje složenijih softverskih mehanizama za distribuiranu koordinaciju.

Zbog ovih dodatnih mehanizama, isti skup hardverskih resursa raspoređen na veći broj instanci ostvaruje nižu efektivnu performansu u poređenju sa konfiguracijom u kojoj se ti resursi nalaze u jednoj fizičkoj instanci. Ovaj efekat dovodi do pojave dodatnih troškova koje uvodi sama distribucija sistema.

Definicija 3.7. *Režijsko opterećenje (eng. overhead) predstavlja potrošnju resursa koju distribuirani informacioni sistem mora da izvrši radi održavanja zajedničke koordinacije, sinhronizacije i upravljanja svojim instancama, a koja nije direktno deo primarne obrade korisničkog zahteva.*

Režijsko opterećenje ima kumulativno negativan uticaj na performanse: svaki novi mehanizam koordinacije povećava vreme obrade pojedinačnog zahteva i umanjuje deo sistema koji može biti posvećen korisničkim operacijama. Kao posledica toga, raste vreme odaziva, dok se efektivna propusnost sistema smanjuje.

Definicija 3.8. *Trošak sinhronizacije (eng. synchronization penalty) predstavlja deo režijskog opterećenja koji nastaje kada čvorovi moraju da razmenjuju svoje trenutno stanje, usklađuju podatke ili čekaju jedni na druge.*

Definicija 3.9. *Kontencija (eng. contention penalty) predstavlja deo režijskog opterećenja nastao usled međusobnog takmičenja čvorova za pristup istom resursu.*

Ovi fenomeni su formalno obuhvaćeni *Univerzalnom zakonu skalabilnosti* [57, 58], koji opisuje kako režijsko opterećenje utiče na promenu **propusnosti** sistema pri povećanju broja instanci.

Teorema 3.3. *Gunterov univerzalni zakon skalabilnosti ima sledeću standardnu formu:*

$$C(N) = \frac{N}{1 + \alpha(N - 1) + \beta N(N - 1)}$$

pri čemu:

$C(N)$ - relativna propusnost pri N čvorova,

N - broj instanci (čvorova),

α - parametar sinhronizacije,

β - parametar kontencije.

Posledica. *Kontencija predstavlja dominantni oblik režijskog opterećenja jer raste kvadratno sa brojem instanci, dok trošak sinhronizacije raste linearno.*

Kako broj instanci raste, sistem može ispoljiti sublinearno povećanje kapaciteta, stagnaciju ili degradaciju performansi [59]. Svaki novi čvor, osim što doprinosi procesnoj snazi, istovremeno uvodi dodatne konflikte u pristupu zajedničkim resursima i zahteva dodatnu koordinaciju, što može rasti brže od korisnog kapaciteta.

3.2.4 Teorija distribuiranih sistema

U distribuiranim informacionim sistemima ne postoji jedinstvena globalna memorija kojoj bi svi čvorovi imali istovremeni i potpuno konzistentan pristup [60]. Umesto toga, podaci su fizički raspodeljeni između više čvorova koji paralelno obrađuju korisničke zahteve. Ovakva organizacija čini globalno napredovanje informacionog sistema izuzetno složenim, jer se lokalne izmene stanja, nastale na jednom čvoru, moraju usaglasiti sa ostalim čvorovima sistema putem razmene poruka.

Definicija 3.10. *Poruka predstavlja elementarnu jedinicu komunikacije između čvorova u distribuiranom sistemu.*

Bez dodatnih mehanizama koordinacije, čvorovi bi razmenom poruka mogli da lokalno napreduju nezavisno jedni od drugih, što bi dovelo do divergentnih verzija podataka i narušavanja konzistentnosti globalnog stanja sistema.

Definicija 3.11. *Koordinacija obuhvata sve mehanizme pomoću kojih distribuirani čvorovi usklađuju svoje napredovanje razmenom poruka, tako da sistem funkcioniše kao jedna logička celina.*

Koordinacija je potrebna svaki put kada lokalna izmena na jednom čvoru treba da bude reflektovana u celom sistemu. Ovaj proces zahteva komunikaciju da su svi čvorovi blagovremeno dobili informaciju o lokalnim promenama. Mehanizam koji omogućava dostavljanje informacija o izmenama svim drugim čvorovima naziva se propagacija [61].

Definicija 3.12. *Propagacija predstavlja proces distribuiranja informacija o promenama sa jednog čvora ka drugim čvorovima u mreži, radi njihovog usaglašavanja.*

Jedan od najrasprostranjenijih mehanizama propagacije u velikim distribuiranim sistemima jeste epidemijski algoritam poznat kao *algoritam ogovaranja* (eng. *gossip*). Algoritmi ogovaranja funkcionišu tako što svaki čvor periodično nasumično bira jedan ili više drugih čvorova i razmenjuje informacije o lokalnim promenama. Visoko su skalabilni, otporni na otkaze i ne zahtevaju centralizovani koordinacioni mehanizam.

Nakon što čvor primi poruku o promeni i uspešno je primeni na svoju lokalnu kopiju podataka, potrebno je potvrditi **inicijatoru** da je operacija zabeležena. Ovaj korak omogućava nadogradnju pouzdanosti i sprečava gubitak ažuriranja.

Definicija 3.13. *Potvrda (eng. acknowledgment) predstavlja poruku kojom čvor signalizira inicijatoru da je uspešno primio, obradio i trajno sačuvao operaciju upisa.*

Propagacija i potvrde čine osnovu šireg mehanizma. Replikacija [62] je fundamentalni koncept koji obezbeđuje da sistem ima više kopija istih podataka na različitim lokacijama i da te kopije ostaju međusobno usaglašene.

Definicija 3.14. *Replikacija predstavlja proces održavanja više kopija (replika) istih podataka na različitim čvorovima distribuiranog sistema, u cilju povećanja dostupnosti i otpornosti na greške.*

Iako replikacija obezbeđuje da svaki čvor poseduje identičan skup podataka, u mnogim sistemima nije neophodno da svi čvorovi održavaju celu bazu. Alternativni pristupi zasnivaju se na razdvajanju podataka na manje, međusobno nezavisne celine.

Definicija 3.15. *Razdeljivanje podataka (eng. sharding) predstavlja tehniku u kojoj se logički skup podataka deli na više disjunktih particija, pri čemu je svaki čvor odgovoran za obradu samo svog podskupa podataka, čime se povećava paralelizam i kapacitet obrade sistema [63].*

Definicija 3.16. *Slojevitost podataka (eng. layering) predstavlja tehniku u kojoj se logički skup podataka organizuje u hijerarhijski raspoređene slojeve, gde osnovni sloj (eng. Layer 1, L1) održava autoritativno stanje sistema, dok viši slojevi obezbeđuju optimizovane, parcijalne ili privremene reprezentacije tog stanja, uz oslanjanje na niže slojeve radi očuvanja konzistentnosti [64].*

U savremenim sistemima najčešće se primenjuju hibridni modeli koji kombinuju različite oblike replikacije, čime se objedinuju njihove komplementarne prednosti i postiže skalabilan, efikasan i otporan distribuirani informacioni sistem.

Kako replikacija omogućava da svaki čvor nezavisno obrađuje korisničke zahteve, a koordinacioni mehanizmi obezbeđuju da sve replike i particije ostanu međusobno usaglašene u meri koju propisuje primenjeni model konzistentnosti, neophodno je sprečiti preopterećenje pojedinačnih delova sistema.

Definicija 3.17. *Raspodela opterećenja (eng. load balancing) predstavlja mehanizam koji dodeljuje dolazne zahteve različitim čvorovima sistema radi optimalnog iskorišćenja resursa, smanjenja vremena odaziva i sprečavanja preopterećenja pojedinačnih instanci [65].*

Međutim, u realnim uslovima pošiljalac ne može unapred znati da li će poruka uspešno stići do odredišnog čvora niti da li će odgovor biti vraćen. Da bi se izbeglo slanje poruka ka nedostupnim instancama, neophodno je pre slanja proveriti njihovu dostupnost.

Definicija 3.18. *Provera ispravnosti čvora (eng. health check) predstavlja mehanizam periodičnog ispitivanja dostupnosti i funkcionalnosti čvorova u distribuiranom sistemu, radi detekcije otkaza i obezbeđivanja da se neuspešne instance blagovremeno isključe iz procesa obrade.*

Iako mehanizmi raspodele opterećenja i provere ispravnosti omogućavaju izbegavanje slanja poruka ka nedostupnim ili preopterećenim čvorovima, sam proces komunikacije između čvorova i dalje ostaje podložan mrežnim kašnjenjima i neizvesnostima u prenosu poruka. Da bi se kvantitativno ocenila efikasnost komunikacije i procenila pouzdanost mreže u realnim uslovima, neophodno je uvesti metriku koja opisuje ukupan vremenski odziv komunikacionog procesa.

Definicija 3.19. *Vreme odaziva (eng. round-trip time, RTT) u distribuiranim mrežama predstavlja kvantitativnu meru koja označava ukupno vreme potrebno da poruka bude poslata od jednog čvora ka drugom i da se odgovor sa odredišta vrati nazad do inicijatora.*

Kako distribuirani sistemi mogu obuhvatati proizvoljno veliki broj čvorova, poruke o izmenama podataka često pristižu iz različitih izvora, u nepredvidivom redosledu i sa različitim kašnjenjima. Da bi replike konzistentno primenile operacije upisa, neophodno je uspostaviti mehanizme koji određuju njihov redosled primene.

Definicija 3.20. *Sat u distribuiranim sistemima predstavlja mehanizam za merenje ili poređenje vremena događaja.*

Distribuirani sistemi ne mogu pretpostaviti postojanje globalnog, savršeno sinhronizovanog sata. Zbog toga se u praksi primenjuju različite vrste satova, od kojih svaki obezbeđuje specifičan model opažanja i poređenja događaja. Umesto oslanjanja na fizički sat, čija se tačnost ne može garantovati usled mrežnih kašnjenja, distribuirani sistemi najčešće koriste logičke časovnike, koji redosled događaja određuju na osnovu njihovih uzročno-posledičnih odnosa i strukture razmene poruka. Iako postoje i drugi modeli merenja vremena, logički časovnici se u praksi izdvajaju kao primarni alat za određivanje redosleda događaja.

Definicija 3.21. *Uređenje (eng. ordering) predstavlja mehanizam kojim se operacijama dodeljuje precizan položaj u redosledu izvršavanja, bilo na svim čvorovima sistema ili na njihovom podskupu [66].*

Samo postojanje uređenja nije dovoljno da sistem funkcioniše ispravno, jer ono određuje isključivo redosled primene operacija, ali ne eliminiše problem paralelnih pristupa nad istim podacima. Istovremeno izvršavanje više operacija nad istim entitetima može dovesti do anomalija, gubitka ažuriranja ili narušavanja logičke konzistentnosti.

Definicija 3.22. *Konflikt predstavlja situaciju u kojoj dve ili više operacija nad istim podacima ne mogu biti istovremeno primenjene bez narušavanja konzistentnosti sistema.*

Zbog toga je neophodno uvesti dodatni mehanizam koji reguliše istovremene pristupe deljenom stanju.

Definicija 3.23. *Kontrola konkurentnog pristupa (eng. concurrency control) predstavlja skup algoritama i tehnika koji obezbeđuju ispravno izvršavanje paralelnih operacija nad deljenim podacima, sprečavajući pojavu konflikata i obezbeđujući konzistentan rezultat [67].*

U praksi se primenjuju dva osnovna pristupa kontroli konkurentnog pristupa:

1. **Pesimistička kontrola** - sprečava konflikte unapred tako što privremeno zaključava podatke pre izvršavanja operacije,
2. **Optimistička kontrola** - dozvoljava paralelne upise, a konflikte otkriva i razrešava prilikom verifikacije ili potvrđivanja rezultata.

3.2.5 Konsenzus

Tokom razvoja distribuiranih sistema predloženi su brojni protokoli za koordinaciju izvršavanja operacija između više učesnika sistema. Rani pristupi zasnivali su se na prisustvu centralnog koordinatora koji komunicira sa svim učesnicima i prikuplja njihove odgovore o mogućnosti izvršavanja određene izmene [68]. Ukoliko **svi** učesnici potvrde spremnost za izvršavanje operacije, izmena se potvrđuje, u suprotnom se odbacuje. Ovakvi protokoli omogućavaju koordinisano izvršavanje distribuiranih transakcija, ali uvode centralnu tačku otkaza u obliku koordinatora.

Kasniji razvoj istraživanja doveo je do nastanka algoritama konsenzusa [69] zasnovanih na razmeni poruka i glasanju između čvorova, što omogućava visok stepen koordinacije u distribuiranim sistemima.

Definicija 3.24. *Konsenzus rešava fundamentalni problem koordinisanog napredovanja stanja u distribuiranim sistemima, u kojem skup čvorova mora da se usaglasi oko jedne zajedničke vrednosti, pri čemu moraju biti zadovoljeni sledeći uslovi:*

- (1) **sporazum** (eng. *agreement*) - svi ispravni čvorovi moraju doneti identičnu odluku,
- (2) **validnost** (eng. *validity*) - odlučena vrednost mora poticati od neke predložene vrednosti,
- (3) **okončanje** (eng. *termination*) - svaki ispravan čvor mora eventualno doneti odluku.

Prvobitni algoritmi konsenzusa omogućavali su postizanje saglasnosti o zajedničkom stanju sistema u prisustvu otkaza pojedinih učesnika (eng. *Crash Fault Tolerance*) [70]. Dalji razvoj distribuiranih sistema doveo je do nastanka algoritama tolerantnih na vizantijske greške (eng. *Byzantine Fault Tolerance*), koji omogućavaju postizanje konsenzusa čak i u prisustvu proizvoljnog ili zlonamernog ponašanja pojedinih učesnika. Među njima se posebno izdvaja *Practical Byzantine Fault Tolerance (PBFT)* [71], koji predstavlja prvi široko prihvaćen praktični algoritam za postizanje konsenzusa u prisustvu vizantijskih grešaka.

Konsenzusni protokoli mogu biti organizovani sa glavnim čvorom (eng. *leader-based*) ili bez glavnog čvora (eng. *leaderless*), u zavisnosti od arhitektonske strukture algoritma. Odlučivanje o pravilnom redosledu i razrešavanje mogućih neslaganja među replikama i dalje obezbeđuje sam konsenzusni protokol, dok glavni čvor služi isključivo kao mehanizam za predlaganje kandidata za narednu odluku. Na taj način glavni čvor pruža stabilan izvor predloga, dok protokol konsenzusa garantuje da sve ispravne replike usvoje isti, jedinstveni rezultat.

U proces konsenzusa ne moraju biti uključeni svi čvorovi u sistemu, već samo određeni podskup čvorova koji čine kvorum (eng. *quorum*). Kako stepen raspodele odlučivanja direktno zavisi od kriterijuma za izbor čvorova koji učestvuju u donošenju odluke, sledi:

Posledica. *Mehanizam formiranja kvoruma u konsenzusnom algoritmu predstavlja ključno mesto kontrolne decentralizacije.*

Zbog posledice iznad, kada postoji fiksni nepromenljivi glavni čvor koji je deo konsenzusnog procesa, može dovesti do smanjenja stepena kontrolne decentralizacije, budući da taj čvor preuzima ulogu centralne tačke koordinacije i serijalizacije predloga operacija.

Nakon uvođenja osnovnih pojmova možemo precizirati tri fundamentalna obrasca propagacije poruka koji se javljaju u realnim distribuiranim sistemima:

1. **Jedan-ka-jedan 1-1** (eng. *one-to-one*) - komunikacija u kojoj jedan čvor upućuje poruku tačno jednom drugom čvoru (npr. slanje izmene glavnom čvoru ili direktna replikacija između dva čvora),
2. **Jedan-ka-više 1-N** (eng. *one-to-many*) - komunikacija u kojoj jedan čvor distribuira poruku većem broju čvorova (npr. glavni čvor replicira zapis svim replikama, ili čvor šalje predlog konsenzus protokolu koji ga prosleđuje svim učesnicima),
3. **Više-ka-više N-N** (eng. *many-to-many*) - komunikacioni obrazac u kojem više čvorova paralelno razmenjuje poruke sa većim brojem drugih čvorova (npr. protokoli bez centralnog koordinatora u kojima čvorovi međusobno razmenjuju poruke radi postizanja saglasnosti).

Važno je istaći da distribuirani sistemi ne moraju nužno posedovati mehanizam konsenzusa. Kada konsenzus postoji, on se tipično primenjuje nad pojedinačnom vrednošću (npr. operacijom, predlogom ili zapisom u logu) o kojoj sistem treba da donese odluku. Svaka odluka konsenzusa predstavlja jedan korak globalnog napredovanja sistema.

Sekvenca konsenzusnih odluka omogućava implementaciju totalno uređenog prenosa poruka (eng. *total order broadcast*) [72], pri čemu svi čvorovi posmatraju isti redosled poruka. Iz toga sledi sledeći rezultat:

Teorema 3.4. *Totalno uređen prenos poruka ekvivalentan je problemu konsenzusa [73].*

Totalno uređen prenos poruka predstavlja globalno svojstvo sistema koje obezbeđuje jedinstven redosled izvršavanja operacija na svim čvorovima. Na osnovu tog redosleda svaki čvor može lokalno izvršavati operacije, bez potrebe za logičkom centralizacijom.

Ukoliko svaki čvor u sistemu beleži sve operacije u istom redosledu, sistem poseduje osobinu replikovanog log-zapisa (eng. *replicated log*) [74]. U prisustvu konsenzusa, jedinstvenost i konzistentnost ovog zapisa su garantovane.

Kada postoji totalno uređen prenos poruka i kada su čvorovi modelovani tako da deterministički izvršavaju operacije u istom redosledu, distribuirani sistem ostvaruje osobinu *replikacije stanja mašine* (eng. *state machine replication*) [75], čime se obezbeđuje jaka konzistentnost sistema.

Međutim, u potpuno asinhronim distribuiranim sistemima javlja se fundamentalno ograničenje izraženo FLP teoremom:

Teorema 3.5. *Ne postoji deterministički konsenzusni protokol koji garantuje okončanje u prisustvu makar jednog neispravnog čvora u potpuno asinhronom sistemu [76].*

S obzirom na to da replikacija stanja mašine zahteva totalno uređen prenos poruka, a time i konsenzus, CP sistemi se modeluju kao parcijalno sinhronizovani sistemi [77], u kojima postoji gornja granica kašnjenja poruka i brzine izvršavanja, ali ona nije unapred poznata.

Sa druge strane, deterministički konsenzus može dovesti do blokiranja sistema u slučaju kašnjenja ili otkaza čvorova, što je u suprotnosti sa principima AP sistema koji prioritizuju dostupnost. U takvim sistemima moguće je koristiti verovatnosni konsenzus (eng. *probabilistic consensus*), u kojem čvorovi iterativno razmenjuju informacije sa nasumično odabranim podskupom drugih čvorova i ažuriraju svoja stanja prema određenom verovatnosnom pravilu. Verovatnoća neusaglašenosti opada, tipično eksponencijalno, sa brojem iteracija [78].

Definicija 3.25. *Verovatnosni konsenzus je nedeterministički mehanizam postizanja dogovora u distribuiranim sistemima koji koristi slučajne procese, pri čemu verovatnoća zadovoljenja svojstava sporazuma, validnosti i terminacije konvergira ka 1.*

Verovatnosni konsenzus zaobilazi ograničenja FLP teoreme napuštanjem determinističkih garancija [79], čime postaje primenljiv u potpuno asinhronim sistemima.

Ipak, ovakav pristup obezbeđuje samo verovatnosne garancije konzistentnosti, što znači da postoji nenulta verovatnoća privremene neusaglašenosti. Zbog toga verovatnosni konsenzus ne može zadovoljiti stroge zahteve determinističke konzistentnosti, već zahteva dodatne mehanizme za razrešavanje konflikata i postizanje konačne saglasnosti.

Dodatno, u procesu determinističkog konsenzusa postoji praktično ograničenje broja čvorova koji mogu efikasno učestvovati u kvorumu. Sa porastom broja učesnika dolazi do povećanja komunikacionog opterećenja, što, u skladu sa Teoremom 3.3, može dovesti do zagušenja mreže i degradacije performansi sistema, a u krajnjem slučaju i do njegovog blokiranja. Ova ograničenja predstavljaju značajan faktor pri projektovanju skalabilnih distribuiranih sistema.

3.2.6 Analiza troškova sinhronizacije i kontencije

Trošak sinhronizacije u distribuiranim sistemima prvenstveno je određen obimom i strukturom komunikacije potrebne da bi se održalo konzistentno i usaglašeno stanje među čvorovima. Na ovaj trošak najznačajnije utiču obrasci propagacije, posebno 1-N i N-N šemi, budući da generišu veći broj međusobnih razmena i povećavaju zahtev za koordinacijom. Dodatni doprinos potiče od mehanizama potvrđivanja operacija, pri čemu broj komunikacionih rundi i njihovo trajanje direktno zavise od vremena odaziva (RTT). U sistemima koji koriste konsenzusne protokole, ovaj trošak dodatno raste jer svaka odluka zahteva više komunikacionih faza i formiranje kvoruma, čime latencija pojedinačne operacije postaje zavisna od najsporijih učesnika u procesu odlučivanja.

Vrsta replikacije takođe snažno oblikuje trošak sinhronizacije: sinhrona replikacija zahteva da sve replike prihvate operaciju pre njenog završetka, što povećava ukupno usklađivanje, dok asinhrona replikacija omogućava smanjenje ovog troška delegiranjem propagacije u pozadinu. Particionisanje generalno smanjuje sinhronizaciju jer ograničava broj čvorova koji moraju biti uključeni u usklađivanje stanja, ali je njegoa primena ograničena u sistemima koji zahtevaju konsenzus, budući da svaka particija mora sadržati dovoljan broj čvorova za formiranje kvoruma i donošenje odluka.

Uređenje operacija predstavlja jedan od strukturalno najskupljih elemenata sinhronizacije, posebno kada je zahtevano totalno uređenje, budući da ono implicira globalno slaganje o redosledu izvršavanja operacija. Mehanizmi kontrole konkurentnog pristupa dodatno uvode sinhronizacione korake, bilo kroz zaključavanje (pesimistički pristup) bilo kroz verifikaciju konflikata (optimistički pristup). U sistemima zasnovanim na konsenzusu, totalno uređenje se ostvaruje kroz sekvencu odluka, što uvodi dodatnu zavisnost između operacija i ograničava mogućnost paralelnog izvršavanja.

Provera ispravnosti čvorova i raspodela opterećenja doprinose trošku sinhronizacije u manjoj meri, ali povećavaju ukupan režijski saobraćaj sistema, odnosno pozadinsku komunikaciju koja nije deo primarne obrade korisničkih zahteva.

U prisustvu vizantijskih čvorova, trošak sinhronizacije se dodatno uvećava zbog potrebe za višerundnom razmenom poruka, verifikacijom poruka i formiranjem kvoruma otpornog na maliciozno ponašanje.

U celini posmatrano, svaki mehanizam koji zahteva striktno uređenje, rad sa kvorumima ili usklađivanje replika proporcionalno povećava trošak sinhronizacije sa rastom broja čvorova i složenošću protokola.

Trošak kontencije potiče iz situacija u kojima više čvorova istovremeno pokušava da pristupi istom resursu ili istoj logičkoj jedinici podataka, što izaziva konkurenciju, zagušenja i čekanje. Ovaj trošak primarno određuju mehanizmi kontrole konkurentnog pristupa: pesimističke tehnike zaključavanja stvaraju direktne konflikte i blokiranja, dok optimistički pristupi prebacuju teret kontencije na kasniju detekciju i razrešavanje konfliktnih upisa.

Obrasci propagacije utiču na kontenciju uglavnom u slučajevima kada više čvorova generiše konkurentne upise nad istim entitetom, pri čemu N-N komunikacija može dodatno uvećati efektivno zagušenje. Duže trajanje zaključavanja ili konfliktne verifikacije povećava vreme odaziva, što dalje amplifikuje kontenciju. Replikacija povećava kontenciju ukoliko sve replike moraju serijalizovati iste operacije, dok particionisanje tu kontenciju smanjuje razdvajanjem domena pristupa. Mehanizmi uređenja takođe doprinose kontenciji jer totalni redosled implicitno uvodi globalnu serijalizaciju.

Provera ispravnosti čvorova može privremeno smanjiti raspoloživu procesnu snagu i time pojačati pritisak na preostale čvorove, dok raspodela opterećenja ublažava kontenciju ravnomernijom distribucijom zahteva.

Prisustvo vizantijskih čvorova dodatno pojačava kontenciju, jer protokoli moraju da serijalizuju

operacije kroz šire kvorume i produžene faze verifikacije kako bi se eliminisali lažni ili kontradiktorni predlozi.

U celini, kontenciju najviše povećavaju arhitekturni obrasci koji uvode zajedničke kritične sekcije, globalna zaključavanja ili serijalizaciju pristupa istim podacima, pri čemu sistemi zasnovani na konsenzusu predstavljaju poseban slučaj u kojem je takva serijalizacija ugrađena osobina sistema.

3.3 Operativna svojstva

Bezbednosni modeli predstavljaju formalizovane teorijske okvire kojima se sistematski definišu bezbednosni ciljevi, principi i pravila ponašanja informacionih sistema u odnosu na pretnje, neovlašćen pristup i zloupotrebu resursa. U kontekstu distribuiranih informacionih sistema, ovi modeli se primenjuju u okruženjima koja karakterišu autonomni čvorovi, nepouzđani komunikacioni kanali i parcijalni otkazi, što značajno komplikuje ostvarivanje i koordinaciju bezbednosnih svojstava.

Ovi modeli čine osnovu savremenih teorijskih i praktičnih pristupa informacionoj bezbednosti i zasnivanju arhitektura visokog nivoa pouzdanosti. Ključna operativna bezbednosna svojstva [80, 81], koja se u literaturi često objedinuju akronimom *CIAAN*, predstavljaju kvalitativne osobine sistema i uključuju:

1. **Poverljivost** (eng. *confidentiality*) - svojstvo kojim se obezbeđuje da podaci ostanu dostupni isključivo ovlašćenim korisnicima, posebno kada je reč o ličnim, osetljivim ili zaštićenim informacijama,
2. **Integritet** (eng. *integrity*) - svojstvo koje garantuje tačnost, potpunost i neizmenljivost podataka, sprečavajući njihovu modifikaciju od strane neovlašćenih korisnika,
3. **Dostupnost** (eng. *availability*) - svojstvo koje obezbeđuje da resursi i usluge budu trajno i blagovremeno dostupni, uprkos greškama, napadima ili opterećenju sistema,
4. **Autentičnost** (eng. *authenticity*) - svojstvo koje omogućava pouzdanu identifikaciju inicijatora koji šalje ili pruža određenu informaciju,
5. **Neporecivost** (eng. *non-repudiation*) - svojstvo kojim se obezbeđuje da korisnici ne mogu negirati prethodno izvršene radnje, poput slanja, primanja ili potvrde poruka.

U distribuiranim sistemima bezbednosna svojstva ostvaruju se kroz koordinisanu interakciju više čvorova, što podrazumeva dodatne komunikacione i računarske troškove. Za razliku od ostalih bezbednosnih svojstava, dostupnost je pretežno određena arhitektonskim i operativnim karakteristikama sistema i u distribuiranim okruženjima se dominantno oslanja na mehanizme replikacije, zbog čega je detaljno razmatrana u prethodnom poglavlju.

Primena bezbednosnih modela suočava se sa brojnim izazovima, uključujući visoku složenost modernih informacionih sistema, troškove implementacije, organizacioni otpor korisnika, kao i tehnička ograničenja integracije različitih sigurnosnih mehanizama. Prestroge bezbednosne politike mogu značajno umanjiti produktivnost i upotrebljivost sistema, dok previše liberalan pristup povećava rizik kompromitovanja podataka [82]. Savremene bezbednosne arhitekture stoga teže uspostavljanju održive ravnoteže između operativnih potreba i nivoa zaštite.

Međutim operativna svojstva informacionih sistema ne mogu se posmatrati izolovano od njihovih arhitektonskih i performansnih ograničenja, budući da svaki mehanizam za obezbeđivanje bezbednosti, pouzdanosti ili integriteta nužno uvodi dodatne troškove u pogledu latencije, propusnosti i

režijskog opterećenja sistema. Ovaj režijski trošak je naročito izražen u distribuiranim sistemima, gde bezbednosni mehanizmi često podrazumevaju dodatne poruke, kriptografske operacije i sinhronizaciju između udaljenih čvorova.

3.3.1 Kriptografski i poverljivi mehanizmi

U cilju ostvarivanja navedenih operativnih bezbednosnih svojstava i njihove primene u realnim informacionim sistemima, uvode se konkretni tehnički mehanizmi, među kojima kriptografija ima centralnu ulogu u obezbeđivanju poverljivosti, integriteta i autentičnosti podataka. Njena suštinska uloga ogleda se u sprečavanju neovlašćenog pristupa informacijama putem formalno definisanih matematičkih transformacija, uz jasno definisane ciljeve, kao što su sigurnost šifrovanja, otpornosti na kriptanalitičke napade i verifikacije autentičnosti poruka.

Definicija 3.26. *Šifrovanje (eng. encryption) je proces pretvaranja podataka u nečitljiv oblik radi zaštite. Dešifrovanje (eng. decryption) je proces vraćanja šifrovanih podataka u njihov originalni, čitljiv oblik.*

U distribuiranim sistemima, kriptografski mehanizmi ne utiču samo na bezbednost, već i direktno na performanse sistema, jer uvode dodatne računarske operacije i komunikacione korake u svaki tok podataka.

U cilju pojednostavljenja opisa kriptografskih protokola i komunikacionih scenarija, u nastavku se učesnici u razmeni poruka konvencionalno označavaju simboličkim imenima: pošiljalac Alisa (A) (eng. *Alice*), primaoc Boban (B) (eng. *Bob*), prislušivač Eva (E) (eng. *Eve*) i zlonamerni napadač Milena (M) (eng. *Mallory*), čime se pojednostavljuje analiza bezbednosti i formalna specifikacija protokola.

3.3.1.1 Simetrična kriptografija

Prvi praktično definisan simetrični kriptografski algoritam projektovan za računare je „Lucifer” [83], razvijen u IBM-u početkom sedamdesetih godina XX veka kao odgovor na rastuću potrebu za automatizovanom zaštitom podataka u bankarskim i finansijskim sistemima. U Luciferu je eksplicitno uvedena upotreba tajnog ključa, tj. numeričke vrednosti koja kontroliše procese šifrovanja i dešifrovanja. Ovim je postavljen temelj savremenim blok šiframa i identifikovani su prvi problemi privatnosti i sigurnosti u računarskim sistemima [84].

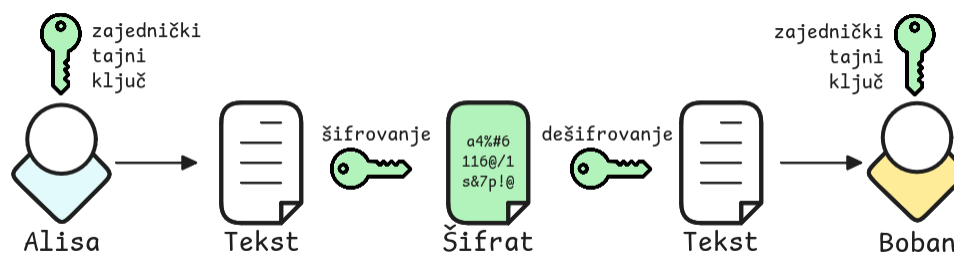
Definicija 3.27. *Simetrična kriptografija predstavlja kriptografski model u kojem se za šifrovanje i dešifrovanje podataka koristi identičan tajni ključ.*

Simetrični kriptografski algoritmi mogu se klasifikovati prema kriptografskoj konstrukciji koju koriste na:

1. **Blok šifre** - obrađuju podatke u blokovima fiksne veličine, primenom višestrukih rundi transformacija, kao što su permutacije, substitucije i mešanje podataka. Ovakav dizajn obezbeđuje visok nivo otpornosti na kriptanalitičke napade. Najzastupljeniji savremeni predstavnici su AES [85] i Camellia [86].
2. **Strim (eng. stream) šifre** - šifruju podatke kontinuirano, na nivou bitova ili bajtova, generisanjem pseudonasumičnog toka podataka koji se kombinuje sa otvorenim tekstom. One su naročito pogodne za aplikacije koje zahtevaju nisku latenciju ili obradu podataka u realnom vremenu. Danas se najčešće koristi ChaCha20 [87].

3. **AEAD (eng. *Authenticated Encryption with Associated Data*) šeme** - koriste postojeće blok ili strim šifre u kombinaciji sa mehanizmima za autentifikaciju čime se istovremeno obezbeđuje poverljivost i integritet/autentičnost podataka. Najkorišćenije AEAD šeme danas su AES-GCM [88] i ChaCha20-Poly1305 [89].

Kada Alisa želi da Bobu prenese poverljivu poruku, ona primenjuje simetrični algoritam šifrovanja koristeći unapred usaglašeni tajni ključ. Nakon što Bob primi šifrat, on primenjuje isti tajni ključ kako bi izvršio dešifrovanje i rekonstruisao originalni sadržaj. Bez pristupa tom ključu, nijedna neovlašćena strana ne može rekonstruisati poruku.



Slika 3.1: Prikaz procesa simetričnog šifrovanja i dešifrovanja u sistemu simetrične kriptografije.

Prednost simetrične kriptografije, pored visokog nivoa sigurnosti, ogleda se u njenoj izuzetnoj brzini i efikasnosti, jer šifrovanje i dešifrovanje troše minimalne resurse i dobro se skaliraju za velike količine podataka. Zbog ovih osobina, simetrična kriptografija predstavlja osnovu za šifrovanje podataka u sistemima sa visokim zahtevima u pogledu propusnosti i niske latencije.

Izazovi simetrične kriptografije uključuju otežano skaliranje sistema sa velikim brojem korisnika, budući da svaki par korisnika mora deliti sopstveni tajni ključ, kao i **problem razmene ključeva** (eng. *key exchange problem*), koji se odnosi na bezbedan prenos tajnog kriptografskog materijala preko nezaštićenog komunikacionog kanala. Tajni ključ mora biti razmenjen na bezbedan način, jer njegova kompromitacija dovodi do ugrožavanja celokupne komunikacije.

U tradicionalnim sistemima to je podrazumevalo da se inicijalna razmena ključeva mora obaviti neposrednim ili prethodno obezbeđenim putem, što predstavlja ozbiljno praktično ograničenje u okruženjima gde se komunikacione strane nalaze na velikim geografskim udaljenostima.

3.3.1.2 Asimetrična kriptografija

Motivisani ograničenjima simetrične kriptografije, predložen je prvi praktični algoritam za uspostavljanje zajedničkog tajnog materijala (eng. *shared secret*) između dve strane tokom inicijalne faze komunikacije preko nezaštićenog kanala [90]. Predloženi mehanizam omogućava da se za svaku novu sesiju nezavisno generiše novi tajni materijal, koji se u savremenim sistemima koristi kao osnova za izvođenje simetričnih privremenih (sesijskih) ključeva.

Pored toga, po prvi put formalno se uvodi koncept javno dostupnih informacija i privatnih tajni, pri čemu bezbednost sistema više ne zahteva postojanje tajnog kanala za inicijalnu razmenu ključeva. Autori takođe uvode teorijsku ideju digitalnih potpisa, čime postavljaju temelj za dalji razvoj kriptografije i savremenih bezbednosnih protokola.

Značajan korak u daljem razvoju predstavlja uvođenje prvog praktičnog asimetričnog kriptografskog algoritma zasnovanog na modularnoj aritmetici, u kome je bezbednost sistema povezana sa računarskom teškoćom faktORIZACIJE velikih kompozitnih brojeva [91]. Ova *asimetrija računarske*

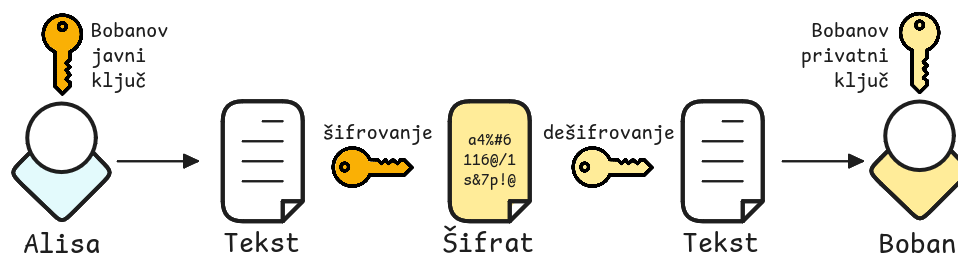
složenosti [92] zasniva se na činjenici da postoje efikasne metode za generisanje i proveru prostih brojeva, poput „Eratostenovog sita” [93], dok je faktORIZACIJA proizvoljnog prirodnog broja na proste činioce izrazito težak problem za koji do danas nije poznat polinomijalno efikasan algoritam [94].

Definicija 3.28. *Asimetrična kriptografija predstavlja kriptografski model u kojem se za šifrovanje i dešifrovanje podataka koriste dva različita, ali matematički povezana ključa: javni ključ, koji može biti slobodno distribuiran, i privatni ključ, koji mora ostati tajan. Poruka se šifrjuje javnim ključem primaoca, dok se dešifrovanje vrši odgovarajućim privatnim ključem, čime se obezbeđuje poverljivost komunikacije.*

Asimetrični kriptografski algoritmi mogu se klasifikovati prema matematičkim problemima na kojima se zasniva njihova sigurnost:

1. **Algoritmi zasnovani na problemu faktORIZACIJE** - Bezbednost ovih sistema zasniva se na teškoći faktORIZACIJE velikih kompozitnih brojeva na proste činioce. Koriste se u okruženjima gde je potrebna kompatibilnost i jednostavnost implementacije. Najistaknutiji predstavnik ove klase je RSA [91].
2. **Algoritmi zasnovani na problemu diskretnog logaritma u multiplikativnim grupama** - Bezbednost ovih sistema zasniva se na teškoći izračunavanja diskretnog logaritma u grupama modularne aritmetike. Ovi algoritmi se danas retko koriste u novim implementacijama zbog većih veličina ključeva. Glavni predstavnici ove klase su Diffie-Hellman protokol razmene ključeva i *Digital Signature Algorithm* [95].
3. **Algoritmi zasnovani na eliptičkim krivama** - Ovi algoritmi koriste algebarsku strukturu eliptičkih krivih nad konačnim poljima i obezbeđuju istu bezbednost kao prethodne klase uz znatno kraće ključeve i bolje performanse. Najzastupljeniji algoritmi su ECDSA [96] (npr. nad krivama secp256k1 [97]) i EdDSA (npr. ed25519 [98]).
4. **Postkvantni algoritmi** - Zasnivaju se na matematičkim problemima za koje nisu poznati efikasni kvantni napadi. Posebno su značajni jer bi Šorov algoritam [99] kompromitovao prethodne tri klase ukoliko se pojave veliki univerzalni kvantni računari.

Kada Alisa želi da Bobu prenese poverljive podatke, ona ih šifrjuje Bobovim javnim ključem, čime obezbeđuje da jedino Bob, koristeći sopstveni privatni ključ, može da dešifrjuje rekonstruiše originalnu poruku.



Slika 3.2: Prikaz procesa šifrovanja i dešifrovanja u sistemu asimetrične kriptografije.

Asimetrični algoritmi značajno su sporiji i računski zahtevniji po pitanju procesorske snage u poređenju sa simetričnim, a za isti nivo bezbednosti zahtevaju znatno duže ključeve. Ovo ograničava

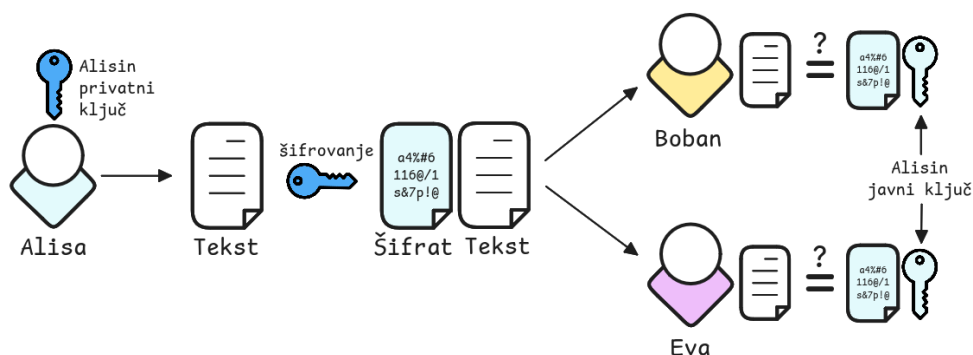
njihovu praktičnu primenu za šifrovanje velikih količina podataka, naročito u resursno ograničenim sistemima. Zbog značajnog performansnog opterećenja, asimetrična kriptografija nije pogodna za kontinuirano šifrovanje podataka u distribuiranim sistemima sa velikim obimom saobraćaja.

Glavna prednost asimetrične kriptografije jeste što eliminiše problem bezbedne distribucije tajnog ključa, koji predstavlja fundamentalno ograničenje simetrične kriptografije. Svaki korisnik poseduje jedinstveni par ključeva, što omogućava skalabilnost u velikim distribuiranim sistemima.

Dodatno, asimetrična kriptografija omogućava digitalne potpise, čime obezbeđuje autentičnost, integritet i neporocivost, što nije ostvarivo isključivo simetričnim pristupom.

Definicija 3.29. *Digitalni potpis predstavlja kriptografski mehanizam u kojem se privatni ključ koristi za generisanje proverljivog potpisa nad porukom, dok se odgovarajući javni ključ koristi za verifikaciju tog potpisa. Proces verifikacije potvrđuje autentičnost pošiljaoca, integritet poruke i obezbeđuje neporocivost autorstva.*

Kada Alisa generiše digitalni potpis korišćenjem sopstvenog privatnog ključa, svako ko poseduje njen javni ključ može da verifikuje potpis i potvrdi da poruka zaista potiče od Alise. Na taj način ostvaruju se autentifikacija pošiljaoca, zaštita integriteta poruke i neporocivost autorstva.



Slika 3.3: Prikaz procesa digitalnog potpisa u sistemu asimetrične kriptografije.

U praksi, savremeni algoritmi digitalnog potpisa najčešće generišu potpis nad hešom poruke, a ne nad samom porukom, čime se postiže veća efikasnost bez promene osnovnog principa rada.

Da bi se obezbedila neporecivost i sprečilo naknadno poricanje ili umetanje događaja u prošlost, javlja se potreba za povezivanjem digitalnih potpisa sa preciznim vremenom njihovog nastanka. U ovom kontekstu razvijen je koncept kriptografskog vremenskog obeležavanja (eng. *time-stamping*) digitalnih dokaza [100], pri čemu se, radi pojednostavljenja modela, pretpostavlja postojanje centralnog autoriteta odgovornog za verifikaciju potpisa i održavanje pouzdane vremenske evidencije.

3.3.1.3 Hibridna kriptografija

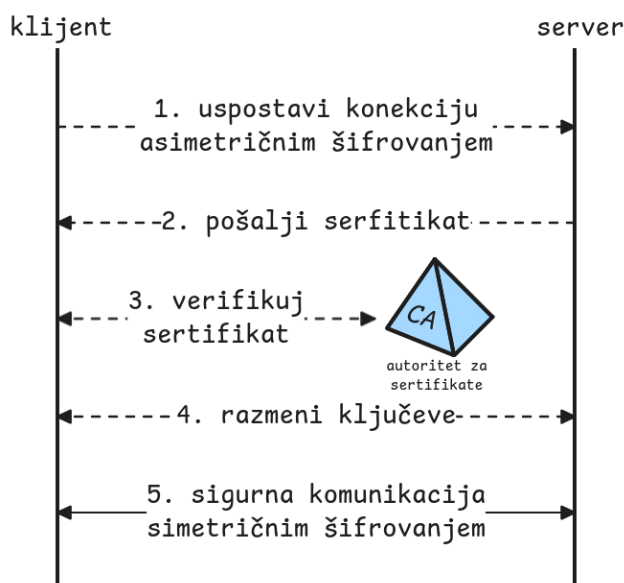
Ideja kombinovanja asimetričnih algoritama za uspostavljanje zajedničkog tajnog materijala i simetričnih šema za efikasno šifrovanje podataka prvi put se implicitno pojavljuje u radu Merkle-a [100], koji se u literaturi često smatra konceptualnom pretečom savremenih hibridnih kriptografskih protokola. Ovaj pristup omogućava da se performansni trošak asimetričnih mehanizama ograniči na inicijalnu fazu uspostavljanja sesije, dok se dalja komunikacija odvija korišćenjem brze i efikasne simetrične kriptografije.

Definicija 3.30. Hibridna kriptografija predstavlja pristup u kojem se za svaku pojedinačnu konekciju (sesiju) najpre uspostavlja glavni zajednički tajni materijal (eng. *shared secret*) korišćenjem asimetričnih algoritama za razmenu ključeva, nakon čega se celokupna dalja komunikacija šifrjuje simetričnim algoritmima.

Iz uspostavljenog zajedničkog tajnog materijala, primenom algoritama za derivaciju ključeva [101] (eng. *Key Derivation Functions*), generiše se čitav skup sesijskih ključeva (eng. *session keys*) koji se koriste za različite kriptografske potrebe, kao što su šifrovanje, autentikacija poruka, itd.

Međutim, aktivni napadač može presresti i čuvati sav saobraćaj, što dovodi do problema **tajnosti unapred** [102] (eng. *forward secrecy*). Naime, ukoliko bi se u budućnosti kompromitovao glavni tajni materijal uspostavljen tokom početnog dogovora, sve ranije razmenjene poruke mogle bi biti dešifrovane. Stoga moderni protokoli koriste algoritme razmene ključeva koji onemogućuju ovo svojstvo.

Najrasprostranjeniji primer hibridne kriptografije jeste protokol TLS [103] (eng. *Transport Layer Security*), koji obezbeđuje poverljivost, integritet i autentičnost podataka tokom komunikacije između klijenta i servera. Prilikom uspostavljanja veze server klijentu dostavlja *digitalni sertifikat* koji sadrži relevantne verifikacione informacije. Nakon verifikacije sertifikata, klijent i server dogovaraju zajednički sesijski tajni materijal, koji se tokom inicijalne razmene štiti asimetričnom kriptografijom. Po uspešnom završetku ovog procesa, dalja komunikacija odvija se primenom efikasnog simetričnog šifrovanja.



Slika 3.4: Prikaz hibridnog kriptografskog modela u TLS protokolu.

Digitalni sertifikat potpisuje autoritet za sertifikate (eng. *Certificate Authority*), koji predstavlja entitet od poverenja zadužen za povezivanje identiteta servera sa njegovim javnim ključem. Na taj način autoritet za sertifikate postaje centralna tačka poverenja u TLS modelu, jer kompromitacija tog entiteta može ugroziti bezbednost komunikacije.

Kriptografski algoritmi za šifrovanje prvenstveno obezbeđuju **poverljivost** komunikacije. Međutim, poverljivost ne podrazumeva nužno i zaštitu integriteta podataka, pa je za to potrebno uvesti dodatne mehanizme.

3.3.1.4 Kriptografske heš funkcije

Simetrične i asimetrične kriptografske šeme su dvosmerne, odnosno podržavaju procese šifrovanja i dešifrovanja. Kriptografske heš funkcije H ne koriste ključeve i ne omogućavaju rekonstrukciju izvornog podatka iz dobijene vrednosti, što ih čini jednosmernim funkcijama, i to je njihova ključna razlika u odnosu na klasične šeme šifrovanja.

Primena heš funkcije na proizvoljan ulazni podatak generiše deterministički izračunatu vrednost fiksne dužine, tzv. *heš* h , pri čemu dužina izlaza zavisi od konkretne primenjene funkcije.

Definicija 3.31. *Kriptografska heš funkcija je deterministička transformacija koja proizvoljan ulazni niz podataka preslikava u izlaz fiksne dužine.*

S obzirom na to da heš funkcija preslikava proizvoljno velik prostor ulaza u konačan prostor izlaza, teorijski je moguće da različiti ulazi imaju isti izlaz (kolizija). Stoga se bezbedna heš funkcija definiše svojstvima koja moraju biti teško ostvariva za napadača:

1. za isti ulaz u funkcija uvek proizvodi isti izlaz h ,
2. promena jednog bita ulaza rezultira izlazom koji se značajno razlikuje od prethodnog,
3. izuzetno je teško pronaći u_1 i u_2 takve da $u_1 \neq u_2$, a $H(u_1) = H(u_2)$,
4. za dati heš h teško je pronaći bilo koji ulaz u za koji važi $H(u) = h$.

Pored bezbednosti, savremene heš funkcije moraju biti dizajnirane za visoku efikasnost i mogućnost obrade podataka proizvoljne dužine. Njihova efikasnost ih čini pogodnim za primenu u distribuiranim sistemima, gde se heš operacije često izvršavaju nad velikim količinama podataka i u realnom vremenu.

Najčešće korišćene moderne heš funkcije uključuju: porodicu SHA-2 [104] (npr. SHA-256), porodicu SHA-3 [105] zasnovanu na Keccak konstrukciji, kao i RIPEMD-160 [106].

Kada Alisa šalje Bobu fajl proizvoljne veličine, najpre se izračunava heš vrednost sadržaja fajla. U slučaju asimetrične kriptografije, nad dobijenom heš vrednošću se generiše digitalni potpis korišćenjem Alisinog privatnog ključa. Alternativno, u sistemima sa uspostavljenim deljenim tajnim ključem, integritet i autentičnost poruke mogu se obezbediti primenom mehanizma autentikacionog koda poruke [107] (eng. *Message Authentication Code, MAC*). Po prijemu fajla, Bob ponovo izračunava heš vrednost primljenih podataka i verifikuje potpis ili MAC koristeći odgovarajući ključ. Ukoliko se vrednosti podudaraju, obezbeđuje se **integritet** podataka i potvrđuje njihovo poreklo.

3.3.2 Funkcionalni bezbednosni okvir

Raniji informacioni sistemi nisu raspolagali jedinstvenim mehanizmima za upravljanje celokupnim okruženjem, već je svaki uređaj unutar mreže morao biti zasebno konfigurisan i održavan. Identitet korisnika nije bio dosledno verifikovan na svim tačkama pristupa, pravila pristupa bila su gruba i neprecizna, sistemi nisu mogli da vode celovite evidencije o aktivnostima korisnika, a ukupna zaštita zavisila je od individualnih podešavanja svakog pojedinačnog uređaja [108].

Definicija 3.32. *AAA [109] (eng. *Authentication, Authorization, Accounting*) predstavlja bezbednosni okvir za proveru identiteta, ograničavanje pristupa i praćenje upotrebe resursa.*

Originalni AAA model se u literaturi i praksi često smatra izuzetno moćnim i funkcionalno nezamenljivim [110], jer se široko tretira kao minimalna i standardizovana arhitektura za kontrolu pristupa. On omogućava logičku centralizaciju bezbednosnih politika i efikasnu kontrolu pristupa celokupnom informacionom okruženju, što je posebno važno u velikim, složenim i distribuiranim sistemima, budući da smanjuje rizik od neovlašćenog korišćenja resursa i zloupotrebe privilegija.

3.3.2.1 Autentikacija

Autentikacija predstavlja inicijalni korak u procesu kontrole pristupa informacionim sistemima, jer mehanizmi autorizacije i evidencije nemaju praktičnu vrednost ukoliko identitet subjekta nije pouzdano utvrđen.

Definicija 3.33. *Autentikacija je bezbednosni proces kojim se proverava i potvrđuje identitet entiteta (korisnika, sistema ili procesa) koji zahteva pristup resursima informacionog sistema.*

Autentikacioni mehanizmi se klasifikuju [111] kao odgovor na različite bezbednosne pretnje, operativna ograničenja i funkcionalne zahteve informacionih sistema:

1. **Autentikacija zasnovana na znanju (eng. *Knowledge-based*)** oslanja se na dokazivanje identiteta putem informacije poznate isključivo legitimnom korisniku, kao što su lozinke, PIN-ovi i tajne fraze. Ova klasa pogodna je za široku primenu zbog jednostavne implementacije i niskih infrastrukturnih zahteva. Međutim, podložna je napadima kao što su pećanje identiteta [112] (eng. *phishing*) i beleženje pritisaka tastera [113] (eng. *keylogging*), kao i napadima grubom silom (eng. *brute-force*), što značajno ograničava njen bezbednosni nivo.
2. **Autentikacija zasnovana na posedovanju (eng. *Possession-based*)** zasniva se na dokazivanju identiteta posedovanjem fizičkog ili digitalnog objekta, poput pametnih kartica, hardverskih tokena ili kriptografskih ključeva. Ovakav pristup omogućava znatno viši nivo bezbednosti kroz upotrebu kriptografskih mehanizama. Glavna ograničenja ove klase su rizik od gubitka ili krađe autentikacionog sredstva, kao i potreba za njegovim sigurnim skladištenjem i upravljanjem.
3. **Autentikacija zasnovana na biometriji (eng. *Inherence-based*)** realizuje se utvrđivanjem identiteta na osnovu fizičkih ili bihevioralnih karakteristika korisnika, kao što su otisak prsta, prepoznavanje lica ili glas. Ovaj pristup eliminiše potrebu za pamćenjem tajni ili posedovanjem dodatnih objekata, ali uvodi izazove u pogledu privatnosti, etičkih pitanja i otpornosti na napade imitacije, uključujući lažne biometrijske uzorke.
4. **Autentikacija zasnovana na kriptografiji (eng. *Cryptographic-based*)** zasniva se na dokazivanju posedovanja kriptografskog ključa bez njegovog otkrivanja, primenom mehanizama kao što su digitalni potpisi, izazov-odgovor protokoli [114], tokeni koji omogućavaju privremeni i ograničeni pristup resursima [115] (eng. *OAuth*) i protokoli zasnovani na dokazima nultog znanja [116] (eng. *zero-knowledge proofs*).

S obzirom na to da nijedna pojedinačna klasa autentikacije ne obezbeđuje potpunu zaštitu, u savremenim sistemima se primenjuje višefaktorska autentikacija [117] (eng. *Multi-Factor Authentication*, *MFA*), kojom se kombinovanjem dva ili više različitih faktora značajno povećava otpornost sistema na kompromitaciju.

Autentičnost se u praksi ostvaruje kako na nivou identiteta, putem mehanizama autentifikacije, tako i na nivou podataka i poruka, primenom kriptografskih mehanizama za verifikaciju porekla i integriteta.

Autentifikacija primarno doprinosi ostvarivanju **autentičnosti** identiteta subjekata koji učestvuju u radu informacionog sistema, obezbeđujući pouzdano vezivanje identiteta za konkretne zahteve za pristup.

3.3.2.2 Autorizacija

Autorizacija predstavlja naredni korak u procesu kontrole pristupa informacionim sistemima, koji se izvršava nakon uspešne autentifikacije subjekta. Dok autentifikacija odgovara na pitanje *ko* je korisnik, autorizacija određuje *šta* je tom korisniku dozvoljeno da radi unutar sistema.

Definicija 3.34. *Autorizacija je bezbednosni proces kojim se određuju dozvoljene operacije koje autentikovani entitet može izvršiti nad resursima informacionog sistema.*

Kontrola pristupa (eng. *Access Control*) predstavlja u literaturi i praksi nezamenjiv bezbednosni mehanizam za sprovođenje odluka o autorizaciji u informacionim sistemima [118]. Njena osnovna uloga ogleda se u doslednom ograničavanju upotrebe resursa u skladu sa unapred definisanim bezbednosnim politikama, čime se obezbeđuje zaštita sistema od neovlašćenog pristupa, prekoračenja ovlašćenja i zloupotrebe privilegija.

Odluke o autorizaciji ne donose se proizvoljno, već se zasnivaju na apstraktnim modelima kontrole pristupa [119], koji formalno opisuju način dodele, provere i sprovođenja prava pristupa nad resursima informacionog sistema. Ovi modeli omogućavaju sistematično i konzistentno upravljanje privilegijama, nezavisno od konkretne tehničke implementacije. U nastavku su prikazani najčešće korišćeni modeli kontrole pristupa, poređani od konceptualno najmanje restriktivnih ka strožim oblicima kontrole:

1. **Diskreciona kontrola pristupa (DAC)** zasniva se na principu vlasništva nad resursima, gde svaki resurs ima svog vlasnika. Vlasnik ima diskreciono pravo da dodeljuje ili oduzima pristup drugim korisnicima ili grupama korisnika, što ovaj model čini fleksibilnim, ali potencijalno podložnim zloupotrebama.
2. **Obavezna kontrola pristupa (MAC)** predstavlja strožiji i statičniji model, u kome su resursi i subjekti označeni bezbednosnim etiketama i nivoima poverljivosti. Pristup se dozvoljava isključivo na osnovu unapred definisanih pravila i hijerarhije bezbednosnih nivoa, bez mogućnosti diskrecione promene od strane korisnika.
3. **Kontrola pristupa zasnovana na ulogama (RBAC)** može se posmatrati kao hibridni pristup koji kombinuje elemente DAC i MAC modela. U ovom modelu, privilegije se ne dodeljuju direktno korisnicima, već ulogama koje odražavaju njihove funkcije unutar organizacije. RBAC je danas jedan od najrasprostranjenijih modela u praksi zbog dobre ravnoteže između bezbednosti i administrativne jednostavnosti.
4. **Kontrola pristupa zasnovana na atributima (ABAC)** predstavlja fleksibilan i dinamički model u kome se odluke o pristupu donose na osnovu kombinacije atributa subjekta, resursa, okruženja i zahteva. Ovaj model omogućava preciznu i kontekstno zavisnu autorizaciju i posebno je pogodan za složene, distribuirane i dinamične sisteme.

Ukoliko sistem nije adekvatno konfigurisan, napadači mogu identifikovati i iskoristiti njegove ranjivosti kako bi neovlašćeno proširili svoja pristupna prava. Ovakvo ponašanje, poznato kao eskalacija privilegija [120] (eng. *privilege escalation*), omogućava napadaču da stekne viši nivo ovlašćenja od predviđenog. Posledice ovakvih napada mogu varirati od ograničenih funkcionalnih poremećaja do ozbiljnih bezbednosnih incidenata, uključujući kompromitaciju poverljivih podataka, kršenje regulatornih zahteva, narušavanje reputacije sistema i značajne ekonomske gubitke.

Zbog toga, u cilju umanjenja ovakvih napada i povećanja bezbednosti, autorizacioni modeli u praksi često primenjuju princip najmanjih privilegija [121], kojim se entitetima dodeljuju isključivo prava neophodna za izvršavanje njihovih zadataka, čime se značajno smanjuje rizik od kompromitacije poverljivih podataka.

Autorizacija ima ključnu ulogu u ostvarivanju **poverljivosti i integriteta** informacionih resursa, jer precizno definiše i dosledno sprovodi koje su operacije dozvoljene nad određenim podacima i funkcijama sistema.

3.3.2.3 Evidencija

Evidencija predstavlja završnu komponentu funkcionalnog bezbednosnog okvira i obuhvata sistematsko prikupljanje, čuvanje i analizu zapisa o aktivnostima autentifikovanih i autorizovanih entiteta u informacionom sistemu. Dok autentifikacija i autorizacija definišu identitet i dozvole subjekata, evidencija omogućava uvid u način na koji se ta prava realizuju u praksi i na koji način su resursi korišćeni tokom rada sistema.

Definicija 3.35. *Evidencija je bezbednosni proces kojim se beleže, čuvaju i analiziraju informacije o aktivnostima subjekata nad resursima informacionog sistema radi nadzora, revizije i ostvarivanja odgovornosti.*

Primarna uloga evidencije ogleda se u povezivanju odluka o autentifikaciji i autorizaciji sa konkretnim izvršenim radnjama. Ove informacije se najčešće evidentiraju u obliku dnevnika aktivnosti (eng. *logs*), koji čine osnovu za nadzor, detekciju anomalija, forenzičku analizu i reviziju sistema. U okruženjima sa visokim rizicima bezbednosti, dnevnici aktivnosti moraju biti dizajnirani tako da u slučaju kompromitacije sistema ne otkrivaju osetljive podatke koji bi mogli poslužiti napadaču za dalju eksploataciju.

U distribuiranim sistemima, evidencija dodatno komplikuje dizajn sistema, jer zahteva korelaciju zapisa sa više čvorova, često bez globalno sinhronizovanog vremena, što povećava skladišne i procesorske zahteve.

Evidencija se može organizovati [80] u sledeće grupe:

1. **Evidencija autentifikacije** - obuhvata zapise o pokušajima autentifikacije (uspešnim i neuspešnim), korišćenim autentifikacionim mehanizmima, vremenu i trajanju autentifikacione sesije, izvoru zahteva (npr. IP adresi, uređaju, lokaciji), promenama autentifikacionih podataka i detekciji sumnjivih obrazaca.
2. **Evidencija autorizacije** - obuhvata zahteve za pristup resursima, donete autorizacione odluke, reference na politike i pravila kontrole pristupa, promene privilegija i administrativne odluke o dodeli ili oduzimanju prava pristupa.
3. **Ostala evidencija** - obuhvata podatke o aktivnostima i korišćenju resursa, bezbednosnim događajima i incidentima, sistemskim i konfiguracionim promenama, kao i zapise o integritetu samih evidencija.

Nakon kompromitovanja sistema, napadač može pokušati da izmeni postojeće zapise evidencije, što je poznato kao manipulacija dnevnica aktivnosti (eng. *log tampering*), ili da umetne zlonamernne ili lažne zapise u sistem (eng. *log injection*). Ovakve aktivnosti imaju za cilj prikrivanje tragova napada ili narušavanje integriteta evidencije. Zbog toga je neophodno projektovati mehanizme koji omogućavaju detekciju neovlašćenih izmena, kao što su kriptografska zaštita zapisa, kontrola integriteta i analiza anomalija u evidenciji [122].

Evidencija primarno doprinosi ostvarivanju **neporecivosti** izvršenih radnji, dok se kriptografskim mehanizmima obezbeđuju proverljivi i vremenski obeleženi dokazi o aktivnostima subjekata u sistemu.

3.3.3 Bezbednosne pretnje i napadi

Pored napada usmerenih na mehanizme autentikacije, autorizacije i evidencije, informacioni sistemi moraju obezbediti i zaštitu komunikacionih kanala. U suprotnom, napadač može presresti komunikaciju i lažno se predstaviti kao legitimni servis ili korisnik, pri čemu korisnik ostaje u uverenju da direktno komunicira sa pouzdanim entitetom, dok se poruke zapravo preusmeravaju kroz napadača [123, 124, 125]. Ovakvi napadi omogućavaju napadaču da u realnom vremenu prikupi osetljive podatke, kao što su lozinke, podaci naloga i finansijske informacije.

Pored toga, direktni napadi na sistem mogu značajno ugroziti njegov rad, naročito u prisustvu zlonamernog softvera (eng. *malware*). Ovakav softver može usporiti rad sistema, omogućiti neovlašćen nadzor nad aktivnostima korisnika, obezbediti napadaču direktan pristup uređaju, potpuno onemogućiti njegov rad ili se dalje širiti unutar mreže [126].

Iako se savremeni bezbednosni standardi i protokoli kontinuirano unapređuju radi povećanja otpornosti na nove pretnje, značajan broj bezbednosnih incidenata i dalje je posledica ljudskih grešaka, kao što su korišćenje slabih lozinki ili nepažljivo rukovanje osetljivim podacima.

Jedan od najvećih izazova u oblasti bezbednosti mreža predstavlja *socijalni inženjering* [127], koji često predstavlja početnu tačku različitih napada, budući da iskorišćava prirodnu ljudsku sklonost ka poverenju. Ova kategorija napada obuhvata tehnike kao što su pecanje (eng. *phishing*), mamčenje (eng. *baiting*) i lažno predstavljanje (eng. *pretexting*).

Statistički podaci [128, 129] potvrđuju da značajan procenat bezbednosnih incidenata uključuje ljudski faktor, često nezavisno od tehnološkog nivoa zaštite. Ovo naglašava potrebu za kontinuiranom edukacijom korisnika, razvojem bezbednosne kulture i sistemskim pristupom upravljanju ljudskim faktorom, zbog čega se ljudski faktor u literaturi prepoznaje kao jedan od ključnih uzroka kompromitovanja čak i visoko zaštićenih sistema.

Dodatno, informacionim sistemima može biti uskraćena dostupnost usluge usled preopterećenja izazvanog velikim brojem zlonamernih zahteva [130], što je poznato kao napad uskraćivanja usluge (eng. *Denial of Service, DoS*). Ovakvi napadi dovode do zagušenja sistema, pri čemu, u skladu sa Teoremom 3.2, dolazi do prekoračenja kapaciteta obrade zahteva.

U naprednijim scenarijima, napadi mogu biti distribuirani, pri čemu više kompromitovanih uređaja, često geografski rasprostranjenih i inficiranih zlonamernim softverom, koordinisano generišu veliki broj zahteva prema ciljnom sistemu. Ovakav oblik napada poznat je kao distribuirani napad uskraćivanja usluge (eng. *Distributed Denial of Service, DDoS*). Iako primarno utiču na dostupnost sistema i ne moraju nužno izazvati trajnu štetu, ovi napadi mogu dovesti do značajnih operativnih i ekonomskih posledica.

Glava 4

Blokčejn kao infrastrukturni sloj informativnih sistema

Digitalni dokumenti u tekstualnom, audio, slikovnom i video formatu podložni su lakoj i često neprimetnoj izmeni, što nameće problem pouzdanog utvrđivanja vremena njihovog nastanka ili poslednje modifikacije. Ovaj izazov verifikabilnog vremenskog obeležavanja digitalnih podataka u nepouzdanom okruženju rešen je kroz koncepte kriptografskog vremenskog obeležavanja, pri čemu se sadržaj dokumenata najpre kriptografski hešira bez njegovog otkrivanja, a zatim povezuje sa prethodnim zapisima putem heš vrednosti, čime se formira lanac kriptografski zavisnih vremenskih oznaka [131, 132, 133]. Ovakva struktura omogućava detekciju svake naknadne izmene, budući da bi promena jednog zapisa narušila konzistentnost svih narednih veza u lancu. Ključni doprinos ovih radova ogleda se u demonstraciji da se integritet i hronološki redosled podataka mogu garantovati oslanjanjem na kriptografske mehanizme i javno proverljive dokaze, bez potrebe za potpunim poverenjem u centralni autoritet.

Masovno slanje elektronske pošte bez značajnog troška omogućilo je ranu zloupotrebu otvorenih računarskih sistema kroz nepoželjne poruke (eng. *spam*). Kao odgovor na ovaj problem, predloženi su mehanizmi u kojima se izvršavanje operacija od značaja uslovljava rešavanjem računski zahtevnih, ali lako proverljivih zadataka. Ovaj pristup uvodi ekonomsku barijeru pristupu resursima, stvarajući asimetriju u kojoj legitimni korisnici snose zanemarljiv dodatni trošak, dok napadači koji pokušavaju da generišu veliki broj zahteva moraju da plate proporcionalno visoku cenu u vidu procesorskog vremena i potrošnje energije [134, 135]. Ovi radovi formalizuju koncept dokazivanja uloženog rada kao mehanizma za regulaciju ponašanja u distribuiranim sistemima i pokazuju da se bezbednost sistema može postići korišćenjem ekonomskog troška, bez oslanjanja na identitet učesnika ili centralnu kontrolu.

Pokušaji realizacije elektronskog novca sa svojstvima fizičkog gotovinskog novca, kao što su privatnost i nepratljivost transakcija, zasnivali su se na upotrebi slepih potpisa i kriptografskih protokola koji omogućavaju izvršavanje plaćanja bez otkrivanja identiteta korisnika. Međutim, ovi sistemi suočili su se sa fundamentalnim problemom dvostruke potrošnje (eng. *double spending*), odnosno mogućnošću da se isti digitalni token iskoristi više puta [136]. Predložena rešenja uglavnom su se oslanjala na centralizovane entitete koji vode evidenciju o već potrošenim jedinicama, čime se implicitno uvodi potreba za poverenjem u treću stranu [137]. Kasniji predlozi [138, 139] nastojali su da redukuju ulogu centralnog autoriteta, ali nisu u potpunosti razrešili problem koordinacije i po-

stizanja globalne saglasnosti u otvorenom distribuiranom okruženju. Ovi radovi jasno identifikuju dvostruku potrošnju kao centralni tehnički izazov u dizajnu digitalnih monetarnih sistema.

4.1 Geneza blokčejn tehnologije

Na prethodno predstavljenim teorijskim i tehničkim osnovama razvija se blokčejn tehnologija, koja svoju prvu široko prihvaćenu praktičnu realizaciju dobija kroz elektronski sistem za plaćanje *Bitcoin* (Bitcoin) [140, 141]. Osnovna inovacija ovog pristupa ogleda se u *objavljivanju svih transakcija kao javno dostupnih*, čime se napušta tradicionalni model privatnih knjiga evidencije (eng. *ledger*) kakve održavaju centralizovane finansijske institucije i uvodi javno dostupna, distribuirana knjiga evidencije koju zajednički održavaju učesnici mreže.

U odsustvu centralnog autoriteta od poverenja, javlja se fundamentalni problem postizanja saglasnosti o stanju distribuirane knjige evidencije među međusobno nepoverljivim učesnicima mreže, kao i obezbeđivanja napredovanja ovako definisanog informacionog sistema. Rešenje ovog problema ostvareno je kroz *Nakamoto konsenzus* [142], koji integriše niz prethodno nezavisnih kriptografskih i sistemskih koncepata.

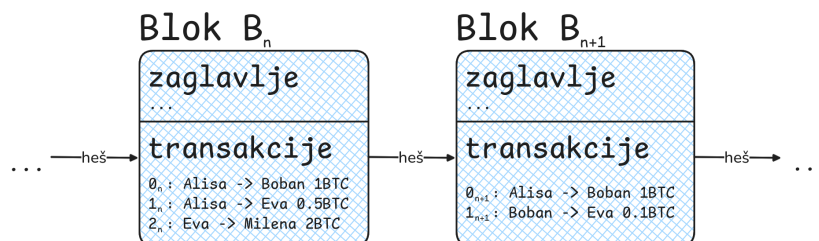
Ovaj mehanizam zasniva se na uvođenju oskudnog resursa (eng. *scarce resource*), čijim dokazivim utroškom učesnici mreže stiču pravo da učestvuju u procesu određivanja narednog stanja sistema. Učesnik čiji predlog bude prihvaćen od strane mreže nagrađuje se prema pravilima protokola. Na taj način tehnički mehanizmi kriptografije dopunjuju se ekonomskim podsticajima, čime nastaje oblast kriptoeconomije (eng. *cryptoeconomics*) [143], u okviru koje se bezbednost sistema ostvaruje usklađivanjem individualnih interesa učesnika sa pravilima protokola.

Definicija 4.1. *Kriptoeconomija predstavlja kombinaciju kriptografskih mehanizama i ekonomskih podsticaja za projektovanje robusnih decentralizovanih informacionih sistema.*

Hronološko povezivanje kriptografski vremenski obeleženih zapisa omogućava formiranje jedinstvenog lanca koji predstavlja evidenciju utroška oskudnog resursa potrebnog za održavanje mreže. Na taj način implicitno se obezbeđuju sigurnost i konzistentnost sistema kroz proverljivu istoriju njegovog napredovanja.

Na ovaj način Bitcoin predstavlja jedinstven, javni i dozvolom neograničen distribuirani sistem koji omogućava postizanje globalne saglasnosti o stanju mreže bez potrebe za centralizovanim autoritetom, uz očuvanje sigurnosti i integriteta podataka.

Definicija 4.2. *Blokčejn predstavlja distribuiranu mrežu ravnopravnih čvorova koja održava zajedničku globalnu knjigu evidencije evidentiranjem transakcija u blokove povezane u strogo hronološki lanac pomoću kriptografskih heš referenci, čime se obezbeđuju nepromenljivost i proverljivost zapisa.*

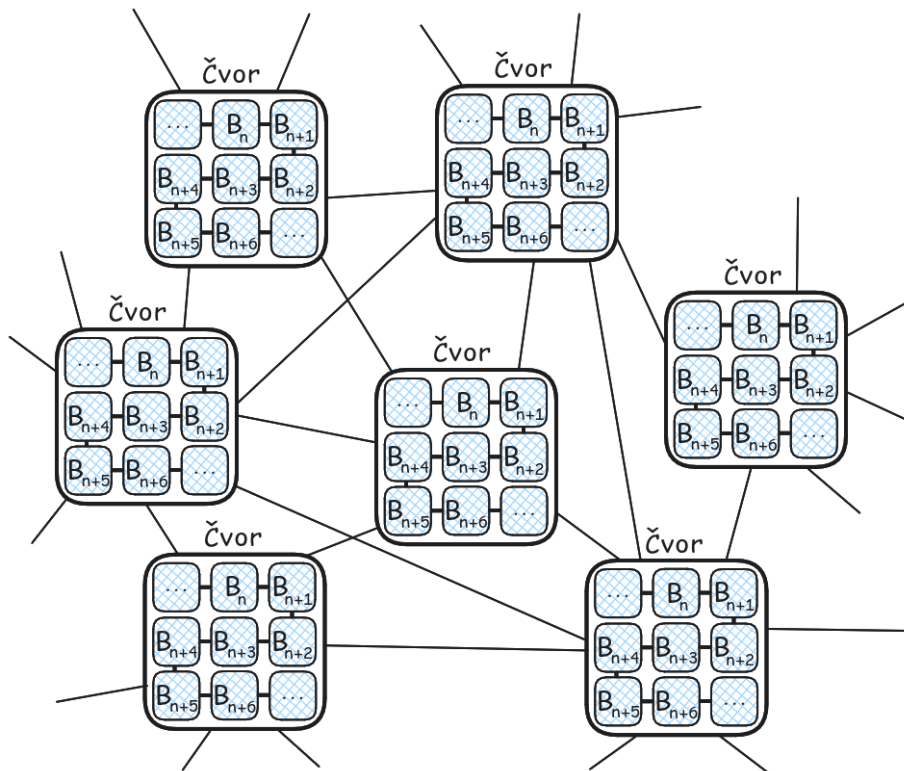


Slika 4.1: Prikaz hronološkog lanca blokova povezanih heš referencama.

Blokčejn sistemi mogu se posmatrati kao posebna klasa distribuiranih sistema u kojima su mehanizmi replikacije, koordinacije i konsenzusa strogo definisani protokolskim pravilima i kriptografskim ograničenjima.

Iako se u izvornom obliku blokčejn definiše kao linearno uređen niz blokova povezanih u strogo hronološki lanac, u kasnijem razvoju pojavile su se i paradigme zasnovane na usmerenim acikličnim grafovima (DAG), koje odstupaju od klasične blokčejn strukture [144]. Ovakvi sistemi zadržavaju određene zajedničke osobine sa tradicionalnim blokčejn mrežama, ali su njihovi mehanizmi konsenzusa, model upravljanja stanjem i struktura podataka suštinski različiti, zbog čega se u daljem tekstu razmatra isključivo klasični blokčejn model zasnovan na linearnoj strukturi blokova.

Budući da se blokčejn posmatra kao distribuirana knjiga evidencije koju zajednički održavaju učesnici mreže, stanje sistema se **replikira** kod svih čvorova. Pri tome, svaki čvor ne samo da održava lokalnu kopiju trenutnog stanja, već i dosledno beleži sva njegova napredovanja u identičnom redosledu u formi lanca blokova. Na taj način blokčejn ostvaruje osobinu **replikovane mašine stanja**, što ga čini **logički centralizovanim** i omogućava determinističku rekonstrukciju stanja sistema u bilo kom trenutku njegove istorije.



Slika 4.2: Distribuirana mreža ravnopravnih čvorova blokčejn mreže sa replikovanim stanjem.

Replikacijom se obezbeđuje da svaki čvor mreže održava lokalnu kopiju distribuirane evidencije, čime se postiže visok stepen **dostupnosti** sistema, kao i otpornost na pojedinačne kvarove i privremenu nedostupnost čvorova. U zavisnosti od arhitektonskih odluka i organizacije sistema, replikacija stanja može biti realizovana u više oblika, među kojima se izdvajaju:

1. **potpuna replikacija stanja**, u kojoj svaki čvor održava celokupno stanje i kompletnu istoriju mreže, u skladu sa Definicijom 3.14, što je karakteristično za većinu klasičnih javnih blokčejn

sistema,

2. **parcijalna replikacija** [145], u kojoj čvorovi održavaju i validiraju samo podskup stanja i transakcija, primenom tehnika razdeljivanja podataka, u skladu sa Definicijom 3.15,
3. **slojevita replikacija** [146], pri čemu se uvode dodatni slojevi kao proširenja iznad osnovnog blokčejna, koji obuhvataju skup protokolskih mehanizama za komunikaciju i verifikaciju stanja u odnosu na osnovni sloj, u skladu sa Definicijom 3.16.

Dodatni slojevi u okviru slojevite arhitekture ne poseduju nezavisan konsenzus mehanizam, već se konačna odluka o validnosti stanja delegira osnovnom lancu, koji ima mogućnost da potvrdi ili ospori rezultate izvršavanja. Iz tog razloga, viši slojevi svoju bezbednost zasnivaju na bezbednosnim garancijama osnovnog sloja. U tom kontekstu, slojevita replikacija uvodi određena ograničenja u pogledu fleksibilnosti, budući da dodatni slojevi moraju biti kompatibilni sa pravilima osnovnog lanca.

Kao alternativni pristup, uvode se *bočni lanci* (eng. *sidechains*) [147], koji omogućavaju uspostavljanje nezavisnih blokčejn sistema povezanih sa osnovnim lancem, bez ograničenja koja nameće slojevita arhitektura, ali uz potencijalnu zavisnost u pogledu vrednosti i dodatni prostor za napade u međulančanoj komunikaciji. Na taj način omogućava se paralelno funkcionisanje raznovrsnih blokčejn sistema sa različitim pravilima izvršavanja i performansnim karakteristikama.

Bez obzira na konkretnu implementaciju, blokčejn sistemi se u literaturi često klasifikuju prema stepenu otvorenosti i modelu kontrole pristupa na dve osnovne kategorije [148]:

1. **Dozvolom neograničeni** (eng. *permissionless*) - *javni blokčejn sistemi* otvoreni za sve učesnike, u kojima ne postoji potreba za prethodnom autorizacijom. U ovim mrežama svaki korisnik može slati transakcije i, uz ispunjenje protokolom definisanih uslova, učestvovati u procesu predlaganja i validacije blokova.
2. **Dozvolom ograničeni** (eng. *permissioned*) - sistemi u kojima je učešće ograničeno na unapred definisan skup identifikovanih i kontrolisanih entiteta. U okviru ove kategorije razlikuju se:
 - *privatni* (eng. *private*) blokčejn sistemi, u kojima jedna organizacija upravlja mrežom i kontroliše proces validacije, čime se ostvaruje centralizovan model upravljanja, i
 - *udruženi* (eng. *consortium*) blokčejn sistemi, u kojima više organizacija zajednički učestvuje u upravljanju i validaciji, ostvarujući delimično decentralizovan model kontrole.

Predviđa se da će budući razvoj blokčejn tehnologije dovesti do okruženja u kojem postoji veliki broj međusobno povezanih, specijalizovanih blokčejn sistema, umesto jedinstvenog dominantnog lanca [149], te će se elektronska trgovina u sve većoj meri proširiti na različite blokčejn platforme [150].

U cilju omogućavanja komunikacije između proizvoljnih blokčejn sistema, uvode se protokoli interoperabilnosti (eng. *interoperability*) [151], koji omogućavaju razmenu vrednosti i podataka između lanaca posredstvom *relejnih* čvorova (eng. *relayers*). Ovi protokoli mogu omogućiti atomično izvršavanje definisano uslovima zapisanim u pametnim ugovorima na oba blokčejna [152].

U zavisnosti od nivoa poverenja i složenosti, često se uvode posrednički *mostovi* (eng. *bridges*) za povezivanje lanaca, koji mogu predstavljati centralnu tačku otkaza i potencijalni vektor napada, uključujući međulančno dvostruko trošenje [153]. Dodatno, povećana složenost ovakvih sistema može negativno uticati na korisničko iskustvo i povećati rizik od implementacionih grešaka koje mogu dovesti do kompromitacije sistema [154].

4.1.1 Transakcije i blokovi

Kada korisnici žele da ostvare interakciju sa mrežom, oni, u skladu sa Definicijom 3.1, formiraju transakciju kao konkretnu instancu elementarnog zahteva sistema, kriptografski je potpisuju odgovarajućim novčanikom i prosleđuju mreži na dalju obradu. U ovom kontekstu, novčanik (eng. *wallet*) predstavlja softverski mehanizam za upravljanje privatnim ključevima ili semenskim frazama (eng. *seed phrase*), kao i za konstruisanje transakcija.

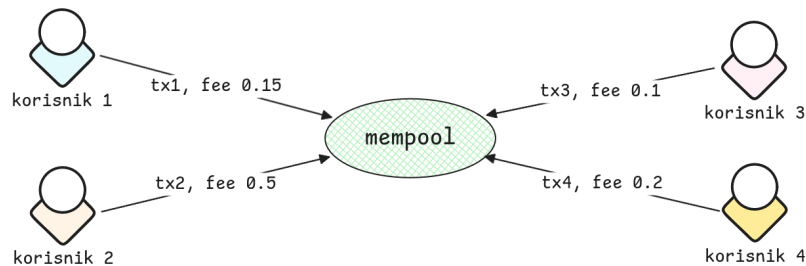
Na ovaj način ostvaruje se *autentikacija zasnovana na kriptografiji*, čime se obezbeđuje **autentičnost** pošiljaoca i dokaz vlasništva nad sredstvima. Istovremeno, upotreba digitalnog potpisa obezbeđuje **neporecivost** transakcije, budući da se validan potpis ne može generisati bez posedovanja odgovarajućeg privatnog ključa.

Iako su transakcije javno vidljive, sistem ne koristi eksplicitne identitete učesnika, već kriptografske adrese, čime se ostvaruje ograničen oblik **poverljivosti** u vidu pseudonimnosti, dok se potpuna zaštita sadržaja transakcija ne obezbeđuje na osnovnom protokolskom nivou.

Definicija 4.3. *Transakcija predstavlja kriptografski zaštićen digitalni zapis kojim se prenosi vrednost ili podaci između učesnika distribuiranog sistema.*

U zavisnosti od načina na koji se potpisane transakcije distribuiraju mreži na dalje procesuiranje, mogu se razlikovati dva osnovna modela blokčejn sistema:

1. Transakcija se smešta u lokalni skup nepotvrđenih transakcija (eng. *mempool*) [155] i propagira kroz mrežu razmenom između čvorova, čime se postiže eventualna konvergencija globalnog skupa nepotvrđenih transakcija, u okviru kojeg čeka da bude uključena u naredni predlog stanja,
2. Transakcija se direktno prosleđuje odgovarajućim čvorovima prema unapred definisanom rasporedu obrade [156], pri čemu se uključuje u naredni predlog stanja bez potrebe za čekanjem u globalnom skupu.



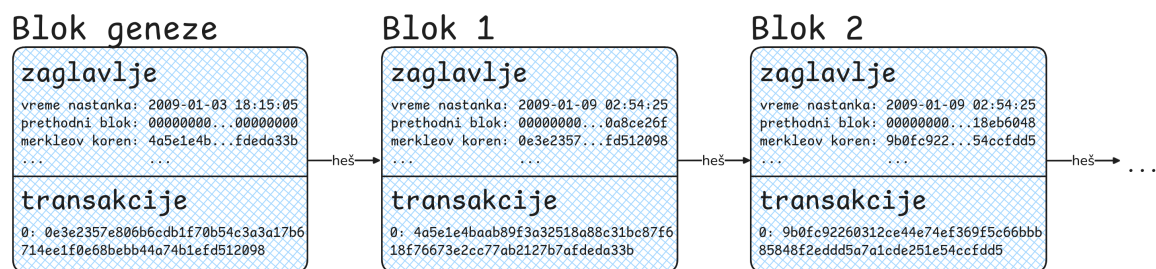
Slika 4.3: Prikaz procesa dostavljanja transakcija korisnika u skup nepotvrđenih transakcija.

U otvorenom i distribuiranom okruženju, obrada, validacija i potvrđivanje transakcija predstavljaju potrošnju zajedničkih računarskih i komunikacionih resursa. Iz tog razloga, javni blokčejn sistemi uvode mehanizam naplate korišćenja mreže primarnom kriptovalutom blokčejna (eng. *native cryptocurrency*), kojim se istovremeno ograničava mogućnost zloupotreba u vidu masovnog slanja beskorisnih transakcija i obezbeđuje ekonomski podsticaj učesnicima koji učestvuju u održavanju i bezbednosti mreže.

Definicija 4.4. *Transakciona naknada (eng. transaction fee, tx fee) predstavlja iznos koji korisnik prilaže uz transakciju kao nadoknadu za korišćenje mrežnih resursa i njeno eventualno uključivanje u blokčejn.*

Transakcije naknade predstavljaju dominantan mehanizam *implicitnog određivanja prioriteta transakcija* u javnim blokčejn sistemima. One stvaraju ekonomski podsticaj za učesnike zadužene za formiranje blokova, koji su motivisani da prilikom selekcije nepotvrđenih transakcija favorizuju one sa većim priloženim naknadama. Na ovaj način ostvaruje se tržišno zasnovan mehanizam raspodele ograničenog kapaciteta blokova, bez potrebe za centralizovanom kontrolom ili eksplicitnim pravilima prioritizacije.

Definicija 4.5. Blok predstavlja skup transakcija i odgovarajuće zaglavlje koje sadrži metapodatke neophodne za verifikaciju i povezivanje blokova, čime se formalizuje globalno napredovanje stanja celokupnog sistema.



Slika 4.4: Prikaz bloka geneze i početnih blokova Bitcoin mreže.

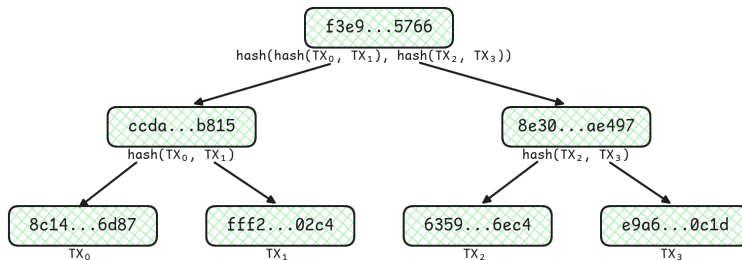
Početni element lanca naziva se *blok geneze* (eng. *Genesis block*) i predstavlja jedini blok koji nema prethodnika, te se eksplicitno inicijalizuje kako bi sistem mogao započeti svoj rad, dok je dodavanje svakog narednog bloka definisano protokolom. Transakcije uključene u blok smatraju se potvrđenim, dok se transakcije unutar istog bloka mogu posmatrati kao logički istovremene, budući da dele istu vremensku oznaku zabeleženu u zaglavlju bloka.

Sve transakcije u okviru bloka organizuju se u Merkleovo stablo, čiji koren predstavlja jedinstvenu heš vrednost koja sažima njihov sadržaj [157]. Svaka izmena bilo koje transakcije dovela bi do promene odgovarajuće heš vrednosti, a time i korena stabla.

Blok 100.000



(a) Zaglavlje i sadržaj



(b) Organizacija transakcija kroz Merkleovo stablo

Slika 4.5: Struktura 100.000. bloka Bitcoin mreže.

Ova vrednost se uključuje u zaglavlje bloka, zajedno sa vremenskom oznakom (koja predstavlja **logički sat** distribuiranog sistema), kriptografskom heš referencom na prethodni blok i dodatnim protokolskim parametrima (verzijom protokola, parametrima konsenzusa i slično).

Heširanjem zaglavlja formira se jedinstvena kriptografska referenca bloka, čime se ostvaruje mehanizam *kriptografskog vremenskog obeležavanja svakog napredovanja sistema*. Ovakva struktura predstavlja osnovu **integriteta** i hronološke konzistentnosti blokčejn sistema, jer bi svaka naknadna izmena sadržaja jednog bloka narušila validnost svih njegovih naslednika u lancu, čime bi bila kompromitovana celokupna sigurnost sistema.

4.1.1.1 Performanse blokčejn sistema

U naučnoj literaturi, performanse blokčejn sistema analiziraju se korišćenjem različitih metrika koje se razlikuju prema nivou apstrakcije i perspektivi posmatranja [158]. U zavisnosti od ciljne grupe i nivoa analize, najčešće se izdvajaju sledeće kategorije metrika:

1. **Makro metrike** - pružaju uvid u performanse sistema iz perspektive krajnjeg korisnika i aplikacionog sloja, posmatrajući sistem kao celinu. Najznačajnije makro metrike obuhvataju propusnost transakcija, izraženu kao broj uspešno potvrđenih transakcija u jedinici vremena, i vreme odaziva, koja predstavlja vreme proteklo od trenutka slanja transakcije do njenog potvrđivanja u sistemu.
2. **Mikro metrike** - odnose se na performanse pojedinačnih komponenti i podprocesa unutar blokčejn sistema, iz perspektive dizajnera i implementatora sistema. Ove metrike obuhvataju, između ostalog, vreme propagacije transakcija i blokova, vreme izvršavanja pametnih ugovora, vreme ažuriranja stanja, trajanje konsenzusnog procesa, kao i efikasnost kriptografskih operacija poput heširanja i digitalnog potpisivanja.

Pored empirijskih merenja, blokčejn sistemi se mogu analizirati i kroz formalne i analitičke modele, sa ciljem izvođenja teorijskih ograničenja performansi bez potrebe za izvršavanjem sistema. Iako performanse obuhvataju više dimenzija, metrika broja transakcija po sekundi često se ističe kao ključni indikator praktične upotrebljivosti sistema, budući da direktno odražava njegov kapacitet za obradu zahteva u realnim uslovima.

Definicija 4.6. *Performanse blokčejn sistema kvantifikuju se kao broj transakcija po sekundi (eng. transactions per second, TPS), što predstavlja dugoročni prosečan broj transakcija koje mreža može uključiti u blokove i potvrditi u jedinici vremena, pod pretpostavkom stabilnog rada sistema.*

Propusnost (eng. *throughput*) blokčejn sistema označava stvarni broj transakcija koje mreža efektivno obradi i potvrdi u jedinici vremena, pri konkretnim uslovima opterećenja, stanja mreže i ponašanja učesnika. Za razliku od kapaciteta, koji predstavlja protokolom definisanu gornju granicu, propusnost je operativna veličina i u praksi može varirati tokom vremena, pri čemu važi da je njena vrednost uvek manja ili jednaka kapacitetu sistema.

U uslovima povećanog intenziteta dolaznih transakcija, kada stopa njihovog pristizanja prelazi kapacitet sistema definisan protokolom, dolazi do **zagušenja**. Kako se broj nepotvrđenih transakcija povećava, raste konkurencija za uključivanje u blok, a s obzirom na to da su transakcije sa višim naknadama prioritetne, dolazi do rasta prosečnih transakcionih naknada, u skladu sa Teoremom 3.2.

Posledica. *Performanse blokčejn sistema su fundamentalno ograničene veličinom pojedinačnog bloka n i očekivanim vremenskim razmakom između generisanja uzastopnih blokova T , koji su određeni protokolom, nezavisno od dodatnih mehanizama kontrole opterećenja ili optimizacija na višim slojevima sistema.*

4.1.1.2 Ograničenja protokola u Bitcoin mreži

U Bitcoinu, zaglavlje bloka ima fiksnu veličinu od 80 bajtova i sadrži skup metapodataka neophodnih za validaciju bloka i njegovo kriptografsko povezivanje sa ostatkom lanca. Ovi metapodaci uključuju:

1. **Verzija** (4 bajta) - identifikator verzije bloka koji određuje skup protokolskih pravila i validacionih uslova koje čvorovi mreže treba da primenjuju prilikom njegove obrade,
2. **Heš prethodnog bloka** (32 bajta) - kriptografska referenca na prethodni blok u lancu, kojom se obezbeđuje da bilo kakva izmena sadržaja prethodnog bloka nužno dovodi do promene zaglavlja tekućeg bloka,
3. **Merkleov koren transakcija** (*tx_root*) (32 bajta) - heš vrednost korena Merkleovog stabla formiranog od svih transakcija uključenih u blok,
4. **Vremenska oznaka** (4 bajta) - trenutak početka procesa heširanja bloka izražen u *Unix* formatu, blok se smatra validnim ukoliko ova vrednost ne odstupa značajno od lokalnog vremena ostalih čvorova u mreži,
5. **nBits** (4 bajta) - kompaktno kodirana reprezentacija **ciljnog praga** (eng. *target hash*), kojom se određuje težina kriptografske zagonetke i dinamički reguliše tempo generisanja novih blokova u mreži,
6. **Počasna promenljiva** (eng. *nonce*) (4 bajta) - slučajna ili iterativno menjana vrednost koja predstavlja rešenje prethodne kriptografske zagonetke, u slučaju iscrpljivanja njenog opsega, dodatno se modifikuje Merkleov koren (eng. *extra nonce*).

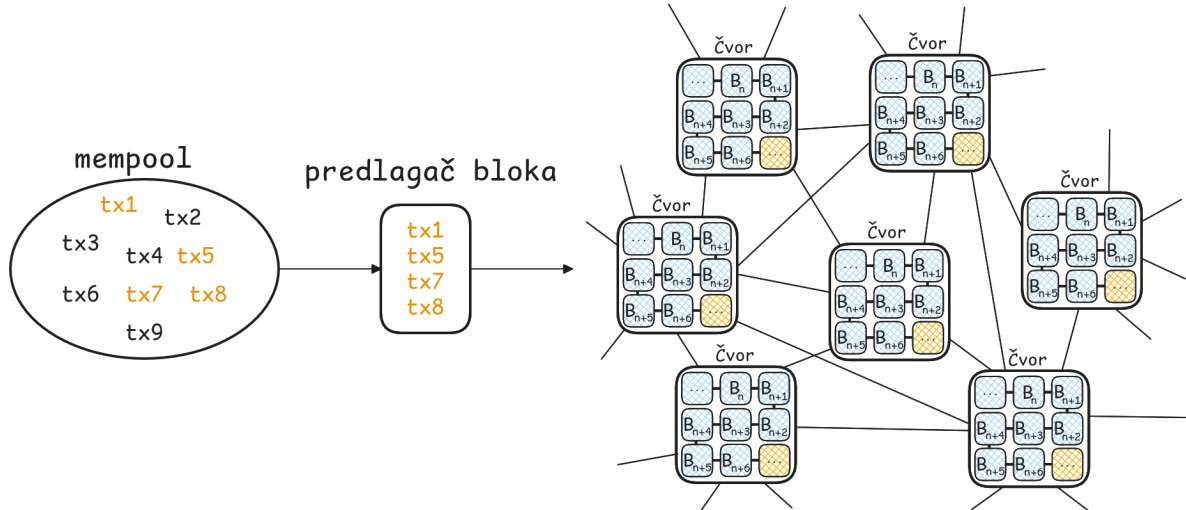
U Bitcoin protokolu, maksimalna veličina bloka je namerno ograničena na 1 MB [159], što predstavlja konzervativan dizajnerski izbor sa ciljem očuvanja decentralizacije i ograničavanja zahteva za skladištenje i propusni opseg čvorova mreže. Ovo ograničenje direktno utiče na performanse i dovodi do efektivnog kapaciteta od približno 7 TPS.

Pokušaji izmene ovog parametra kroz povećanje veličine bloka bili su predmet višegodišnjih tehničkih i ideoloških rasprava unutar zajednice (eng. *Bitcoin scaling debate*), koje su u pojedinim slučajevima rezultovale trajnim podelama mreže i nastankom alternativnih protokolskih implementacija.

4.1.2 Sistem postizanja konsenzusa i validacija novih blokova

Kako bi se formirao kandidat za novi blok, potrebno je najpre prikupiti nepotvrđene transakcije, pri čemu je maksimalna veličina bloka unapred definisana protokolskim parametrima. U blokčejn sistemima, svaki učesnik koji ispunjava uslove u pogledu posedovanja oskudnog resursa definisanog protokolom, poput posedovanja odgovarajućih računarskih resursa i/ili primarne kriptovalute sistema, može pokrenuti instancu sistema i učestvovati u procesu predlaganja i validacije blokova. Implementacije blokčejn protokola realizuju se kroz različite klijentske softvere, koji se međusobno razlikuju prema arhitektonskim pristupima, pri čemu su najzastupljeniji monolitni i višeslojni modeli.

Definicija 4.7. *Predlagač bloka (eng. block proposer) je učesnik mreže koji, u skladu sa pravilima konsenzus protokola, ima pravo da predloži sledeći blok u lancu.*



Slika 4.6: Prikaz propagacije novog bloka kroz mrežu nakon njegovog predlaganja.

U blokčejn sistemima polazi se od pretpostavke da deo učesnika mreže može ispoljavati proizvoljno ili zlonamerno ponašanje, odnosno da sistem mora biti otporan na prisustvo **vizantijskih čvorova**. Iz tog razloga, konsenzus mehanizam predstavlja centralnu tačku **koordinacije** i **kontrolne decentralizacije**, jer definiše skup pravila prema kojima se obezbeđuje konzistentnost globalnog stanja mreže u uslovima nepoverenja među učesnicima.

Osnovno pravilo podrazumeva da se svaki predloženi blok nezavisno proverava od strane svih čvorova pre njegovog prihvatanja i uključivanja u lokalnu verziju lanca. Ovaj proces validacije obuhvata proveru ispravnosti svih transakcija i kriptografskih uslova sadržanih u bloku, čime se eliminišu pokušaji vizantijskih čvorova da nametnu nevalidne ili nekonzistentne promene stanja sistema. Iz toga sledi da je korektno funkcionisanje sistema uslovljeno pretpostavkom da većina resursa koji učestvuju u konsenzusu pripada poštenim učesnicima mreže.

Definicija 4.8. *Konsenzus mehanizam u blokčejn sistemima predstavlja skup protokola i pravila kojima se, u prisustvu potencijalno vizantijskih učesnika, obezbeđuje koordinisano prihvatanje novih blokova i održavanje jedinstvenog stanja distribuirane evidencije.*

U zavisnosti od načina rešavanja izbora predlagača bloka, tolerancije grešaka i postizanja saglasnosti, konsenzus mehanizmi u blokčejn sistemima [160] mogu se klasifikovati u tri osnovne kategorije:

1. **Nakamoto konsenzus mehanizmi**, koji omogućavaju otvoreno i dozvolom neograničeno učešće u procesu predlaganja blokova uvodeći oskudan resurs, oslanjaju se na verovatnosno određivanje predlagača bloka i obezbeđuju verovatnosnu finalnost jer može doći do račvanja (eng. *fork*) blokčejna,
2. **Konsenzus mehanizmi sa tolerancijom vizantijskih grešaka** (eng. *Byzantine fault tolerance, BFT*), koji se zasnivaju na eksplicitnom glasanju unutar unapred definisanog skupa validatora u granicama dozvoljenog broja vizantijskih čvorova i obezbeđuju determinističku finalnost, čime se eliminiše mogućnost trajnih račvanja,
3. **Hibridni konsenzus mehanizmi**, koji kombinuju elemente Nakamoto-stil i BFT pristupa, tipično koristeći verovatnosni mehanizam za proizvodnju blokova i BFT protokole za postizanje konačne finalnosti.

Opisani proces nezavisne validacije, koordinacije i replikacije svakog novog bloka kod svih čvorova mreže predstavlja značajan izvor **režijskog opterećenja** blokčejn sistema. Ovo režijsko opterećenje obuhvata, između ostalog, kriptografsku proveru transakcija, verifikaciju konsenzusnih uslova, razmenu poruka neophodnih za koordinaciju među čvorovima, kao i propagaciju blokova kroz mrežu i održavanje konzistentne lokalne kopije lanca. Iako nijedna od navedenih aktivnosti neposredno ne doprinosi izvršavanju korisničkih transakcija, one predstavljaju nužan preduslov za očuvanje sigurnosti, konzistentnosti i integriteta distribuirane evidencije.

Posledica. *Režijsko opterećenje javnih blokčejn sistema predstavlja sistemski trošak decentralizovanog modela rada, izražen kroz dodatnu potrošnju resursa koja je neizbežna posledica tolerancije vizantijskih grešaka i odsustva centralnog autoriteta od poverenja.*

4.1.2.1 Algoritam dokazivanja uloženog rada

Algoritam dokazivanja uloženog rada (eng. *Proof of Work, PoW*) zasniva se na rešavanju kriptografske zagonetke definisane u bloku na trenutnom vrhu lanca, pri čemu je zagonetka konstruisana tako da poseduje osobinu *asimetrije računarske složenosti*, odnosno da pronalaženje rešenja bude računski zahtevno, dok je njegova verifikacija efikasna i deterministička.

Da bi novi blok bio prihvaćen, njegova heš vrednost mora biti manja ili jednaka unapred definisanom ciljnom pragu, čime se omogućava otvoreno učešće u procesu predlaganja novih blokova bez oslanjanja na centralizovani autoritet ili unapred dodeljeni identitet.

U Bitkoin mreži, oskudan resurs na kojem se zasniva Nakamoto konsenzus predstavlja računarska snaga koja se meri brojem izvršenih heš operacija u jedinici vremena (eng. *hashpower*), koja se u praksi najčešće izražava u terahašima u sekundi (10^{12}). Ova proces zahteva značajnu količinu električne energije i procenjuje se da konzumira oko 0.43% globalne godišnje potrošnje električne energije [161].

Zbog značajnog računarskog i energetskog napora potrebnog za izvršavanje ovog procesa, učesnici koji obavljaju ovu funkciju nazivaju se rudari (eng. *miners*), pri čemu je verovatnoća da određeni rudar predloži naredni blok može se izračunati kao odnos njegove računarske snage, izražene u heševima, prema ukupnoj računarskoj snazi mreže, takođe izraženoj u heševima.

Svaki rudar nezavisno konstruiše kandidat za novi blok, pri čemu samostalno bira skup transakcija koje će u njega uključiti. Tokom izvršavanja algoritma dokazivanja uloženog rada, jedina promenljiva koja se sistematski menja u zaglavlju bloka jeste *nonce*, dok je dominantna i računski najzahtevnija operacija višestruko kriptografsko heširanje kompletnog zaglavlja bloka, koje se vrši sve dok se ne dobije rezultat koji zadovoljava postavljeni uslov (Algoritam 4.1).

Algoritam 4.1 Proces rudarenja

```
1: while currentHash > targetHash do
2:   currentHash = hash(...block header, nonce)
3:   if currentHash ≤ targetHash then
4:     Propagiraj blok
5:   else
6:     Uvećaj nonce
7:   end if
8: end while
```

Ova fundamentalna asimetrija između visokog troška generisanja dokaza i zanemarljivog troška njegove verifikacije predstavlja jedno od ključnih svojstava zbog kojih je računarska snaga izabrana

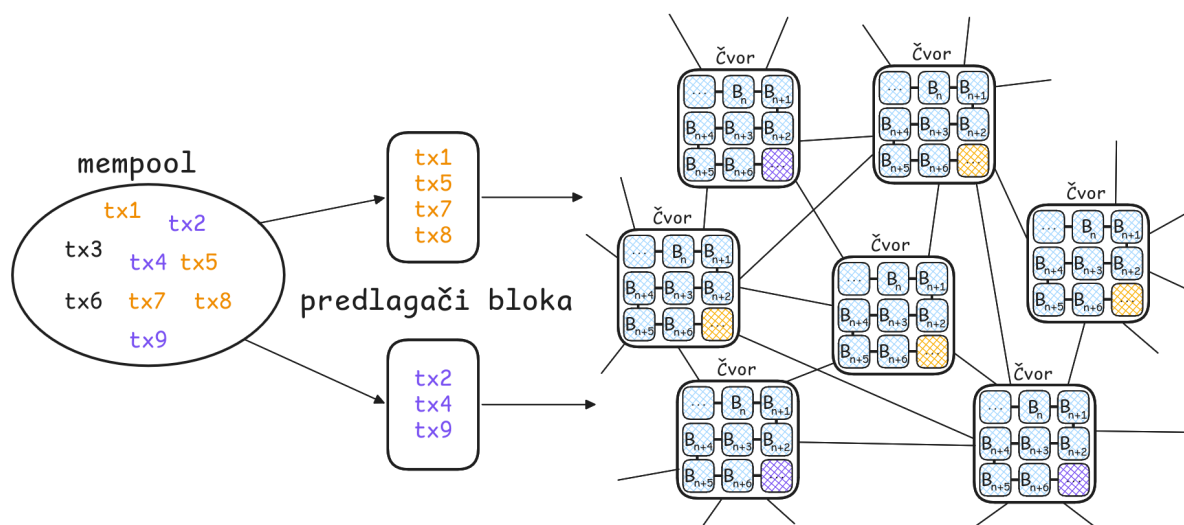
kao oskudan resurs Nakamoto konsenzusa, omogućavajući bezbedno funkcionisanje sistema u otvorenom i nepouzdanom distribuiranom okruženju.

Kako bi se očuvalo približno konstantno prosečno vreme generisanja bloka od deset minuta, Bitcoin protokol periodično prilagođava težinu ciljne heš vrednosti u zavisnosti od ukupne heš snage dostupne u sistemu. Ovo prilagođavanje se vrši na svakih 2016 blokova, čime se omogućava stabilno funkcionisanje sistema uprkos promenama u broju i kapacitetu učesnika koji učestvuju u procesu rudarenja.

4.1.2.2 Verovatnosni konsenzus i praktična finalnost transakcija

Nakon pronalaska odgovarajuće vrednosti *počasne promenljive* predloženi blok se **propagira** ostalim čvorovima u mreži, koji nezavisno proveravaju validnost dokaza uloženog rada i usklađenost bloka sa protokolskim pravilima. Ukoliko su svi uslovi zadovoljeni, blok se prihvata kao naredni blok u lancu, čime se ostvaruje **replikacija** globalnog stanja kod svih učesnika.

Moguće je, međutim, da dva ili više predlagača blokova u približno istom vremenskom intervalu pronađu i propagiraju različite, ali međusobno validne kandidate za $(n+1)$ -ti blok koji se nadovezuju na isti prethodni n -ti blok. U tom slučaju, različiti podskupovi čvorova mogu privremeno prihvatiti različite verzije lanca kao važeće, što dovodi do kratkotrajnog račvanja stanja mreže u više verzija, čime se javlja privremeni **konflikt** u distribuiranom okruženju. Ovo stanje se razrešava pronalaženjem narednog $(n+2)$ -og bloka, budući da se svaki novi blok može nadovezati isključivo na jedan od postojećih kandidata.



Slika 4.7: Prikaz konflikta nastalog usled istovremenog predlaganja više blokova.

U Bitcoin sistemu ne postoji centralni čvor, međutim, pravilo najdužeg lanca (eng. *longest chain rule*) Nakamoto konsenzusa efektivno preuzima njegovu ulogu, budući da je za njegovu izgradnju utrošeno najviše računarskih resursa za algoritam dokazivanja uloženog rada. Grana lanca koja se prva produži i postane duža od ostalih prihvata se kao kanonska verzija istorije sistema. Čvorovi koji su prethodno sledili kraću granu odbacuju je i sinhronizuju se sa najdužim važećim lancem, čime se mreža ponovo dovodi u konzistentno stanje. Kako se nad određenim blokom nadograđuje sve veći broj novih blokova, verovatnoća da alternativna grana nadmaši postojeći lanac opada eksponencijalno.

Definicija 4.9. *Finalnost (eng. finality) transakcije označava trenutak nakon kojeg se transakcija smatra trajno zabeleženom u blokčejnu i ne može biti poništena, čime se ostvaruje praktična neporecivost.*

Čvorovi koji su prethodno sledili kraću granu prelaze na najduži važeći lanac kroz proces reorganizacije (eng. *re-org*). Transakcije koje su bile uključene u blokove odbačene grane, a koje se ne pojavljuju u prihvaćenom lancu, ponovo se smatraju nepotvrđenim i vraćaju u proces daljeg distribuiranja i validacije, pri čemu prolaze istu proceduru kao i sve nove transakcije.

Opisani mehanizam dokazivanja uloženog rada pokazuje da Bitcoin ne postiže konsenzus u strogo determinističkom smislu, već u obliku verovatnosne saglasnosti. U praksi se transakcija smatra prihvaćenom kada se nalazi u najdužem lancu i kada je nad njenim blokom izgrađen dovoljan broj narednih blokova, *najčešće šest*. Ovo pravilo predstavlja inženjerski kompromis između vremena čekanja i prihvatljivog nivoa rizika, pri čemu je verovatnoća reorganizacije lanca nakon tog broja **potvrda** zanemarljiva u realnim uslovima rada mreže.

Nakamoto konsenzus obezbeđuje da je verovatnoća uspešne reorganizacije lanca od strane napadača uslovljena količinom kumulativnog rada koji bi morao da nadoknadi. Konkretno, napadač bi morao da ponovi dokaz uloženog rada za ciljani blok i sve njegove naslednike, a zatim da sustigne i nadmaši ukupni rad poštenih čvorova. Kako broj blokova koji se nadovezuju na posmatrani blok raste, verovatnoća uspeha takvog napada opada eksponencijalno.

Interval između podnošenja transakcije u skup nepotvrđenih transakcija i njenog dostizanja praktične finalnosti može se posmatrati kao **vreme odaziva** sistema, koje takođe zavisi od opterećenja mreže i visine transakcione naknade.

4.1.3 Struktura i funkcionalnost blokčejn sistema

Razumevanje unutrašnje organizacije podataka u blokčejn sistemima predstavlja ključni preduslov za analizu njihovog funkcionisanja i bezbednosnih svojstava. Podaci ne obuhvataju isključivo evidenciju istorije izvršenih transakcija, već i reprezentaciju trenutnog stanja mreže, odnosno način na koji se stanje sredstava i prava pojedinačnih učesnika definiše, ažurira i verifikuje. Različiti modeli upravljanja stanjem uslovljavaju načine potrošnje sredstava, pravila validacije transakcija, kao i skup funkcionalnosti koje sistem može ponuditi krajnjim korisnicima.

Jedan od fundamentalnih bezbednosnih problema koje svaki blokčejn sistem mora da reši na protokolskom nivou jeste *problem dvostruke potrošnje* sredstava. U decentralizovanom okruženju bez centralnog autoriteta, neophodno je obezbediti da se ista jedinica vrednosti ne može validno potrošiti više puta. U blokčejn arhitekturama ovaj problem se rešava kroz dva dominantna modela upravljanja stanjem (eng. *data model*), koji direktno određuju način organizacije globalnog stanja:

1. **Model nepotrošenih izlaza transakcija** (eng. *Unspent Transaction Output, UTXO*), globalno stanje predstavlja skup nepotrošenih izlaza transakcija, pri čemu se potrošnja svakog izlaza vrši u celosti, što omogućava ugrađenu zaštitu od dvostruke potrošnje,
2. **Model globalnog stanja zasnovan na nalogima** (eng. *account-based model*), globalno stanje predstavlja mapu naloga realizovanu odgovarajućom strukturom podataka, pri čemu se za svaki nalog vodi brojač izvršenih transakcija na protokolskom nivou (eng. *protocol-level*), ili se kontrola izvršavanja implementira na nivou pametnih ugovora (eng. *contract-level*), čime se sprečava dvostruka potrošnja

Pored osnovne podele prema modelu upravljanja stanjem, oba pristupa se dodatno mogu klasifikovati prema načinu izvršavanja transakcijske logike, odnosno prema stepenu sinhronizacije izvršavanja:

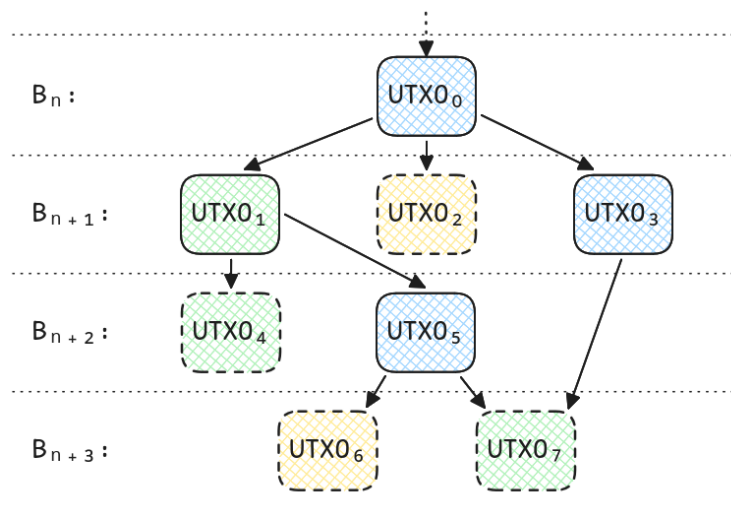
1. **Sinhroni modeli**, u kojima se celokupna transakcijska logika izvršava atomično u okviru jedne transakcije, bez generisanja odloženih poruka, budućih poziva ili sporednih efekata koji bi se realizovali u kasnijim blokovima,
2. **Asinhroni modeli**, u kojima izvršavanje transakcije može proizvesti poruke, događaje ili pozive ka drugim ugovorima ili adresama, čije se izvršavanje odvija u jednom od narednih blokova.

4.1.3.1 Bitcoin Script kao mehanizam kontrole pristupa

Bitcoin sistem koristi sinhroni UTXO model za predstavljanje i praćenje stanja sredstava u mreži. Po uspešnom dodavanju novog bloka u blokčejn, protokol automatski generiše posebnu transakciju bez ulaznih UTXO referenci, kojom se nova sredstva, zajedno sa prikupljenim transakcionim naknadama iz tog bloka, dodeljuju adresi učesnika koji je predložio i uspešno dodao novi blok. Ova transakcija, poznata kao *blagajnička transakcija* (eng. *coinbase transaction*), predstavlja jedini mehanizam kreiranja novih jedinica kriptovalute u obliku UTXO zapisa u Bitcoin sistemu, prema unapred definisanom monetarnom rasporedu.

Definicija 4.10. U okviru UTXO modela, svaka transakcija troši jedan ili više postojećih UTXO zapisa kao ulaze (eng. *inputs*) i generiše nove UTXO zapise kao izlaze (eng. *outputs*).

Budući da se ulazi transakcije odnose na prethodno kreirane, ali još uvek nepotrošene izlaze, prilikom formiranja transakcije korisnik mora eksplicitno navesti koje UTXO zapise namerava da potroši kao ulaze, kao i nove UTXO zapise koje transakcija generiše na svojim izlazima. Iz tog razloga UTXO zapisi se jednoznačno identifikuju kombinacijom identifikatora transakcije u kojoj su nastali (*tx_hash*) i indeksa izlaza (*vout*).



Slika 4.8: Prikaz nastanka i potrošnje UTXO zapisa kroz blokove, pri čemu boje označavaju korisnike, dok pune i isprekidane linije predstavljaju potrošene, odnosno nepotrošene UTXO zapise.

Ukoliko korisnik želi da prenese samo deo vrednosti raspoloživog UTXO zapisa određenoj adresi, on kao ulaz koristi taj UTXO u celosti, dok se na izlazu generišu dva nova UTXO zapisa: jedan u iznosu namenjenom primaocu i drugi koji predstavlja ostatak sredstava i vraća se na adresu pošiljaoca (eng. *change output*).

Slično tome, u slučaju kada vrednost pojedinačnog UTXO zapisa nije dovoljna za izvršenje željene transakcije, korisnik može kombinovati više sopstvenih UTXO zapisa kao ulaze, čime se njihova zbirna vrednost koristi za formiranje proizvoljnog broja izlaza transakcije.

Međutim, broj i struktura ulaznih i izlaznih UTXO zapisa direktno utiču na veličinu transakcije, a samim tim i na količinu prostora koju ona zauzima u bloku. Kako je kapacitet bloka ograničen, transakcije se vrednuju prema broju iskorišćenih virtuelnih bajtova (*vByte*) [162].

Ukupna transakciona naknada izračunava se kao proizvod veličine transakcije izražene u virtuelnim bajtovima i stope naknade po virtuelnom bajtu. Navedena stopa formira se tržišno i menja se u zavisnosti od opterećenja mreže, pri čemu viša stopa naknade povećava verovatnoću da transakcija bude uključena u naredne blokove.

Prilikom trošenja UTXO-a transakciona naknada se implicitno definiše kao razlika između ukupne vrednosti ulaznih i izlaznih UTXO zapisa, što se formalno može izraziti sledećom relacijom:

$$tx_fee = \sum_{x \in UTXO_{in}} value(x) - \sum_{y \in UTXO_{out}} value(y)$$

Ovim se implicitno uvodi uslov da ukupna vrednost ulaza transakcije mora biti jednaka ili veća od zbira vrednosti izlaza.

Ovakvo korišćenje UTXO-a u celosti predstavlja *atomičnu operaciju* kojom se obezbeđuje prevencija dvostruke potrošnje, jer se svaki UTXO, nakon što je iskorišćen kao ulaz i transakcija uspešno validirana, trajno uklanja iz skupa raspoloživih UTXO zapisa.

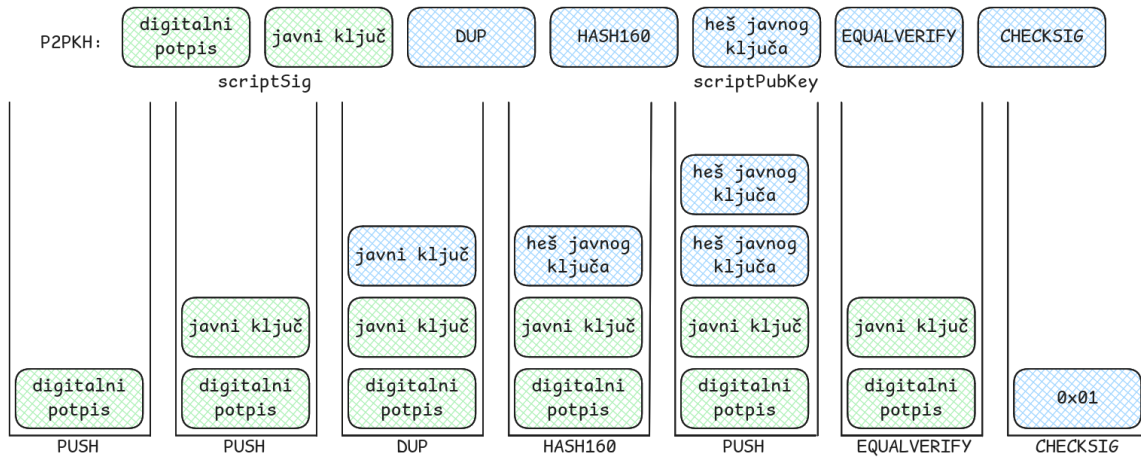
Mehanizam kojim Bitcoin mreža obezbeđuje da samo ovlašćeni korisnik može potrošiti određeni UTXO zasniva se na skriptnom jeziku koji je sastavni deo transakcione strukture, poznatom kao *Bitcoin Script* (Bitcoin Skript). Reč je o jednostavnom, ali izrazito robusnom skriptnom jeziku zasnovanom na steku (eng. *stack-based*), koji je namerno projektovan kao **netjuring-kompletni jezik** bez podrške za petlje i rekurziju, čime se garantuje konačnost izvršavanja svake skripte. Ovakav dizajn omogućava visoku predvidivost i formalnu proverljivost izvršavanja, dok se tokom procesa validacije transakcije putem skripte formalno proveravaju uslovi za potrošnju datog UTXO zapisa.

Ovaj skriptni jezik omogućava sprovođenje **autorizacije** nad sredstvima, budući da definiše skup formalnih uslova koji moraju biti ispunjeni kako bi pristup bio odobren. U tom smislu, može se posmatrati kao pojednostavljen oblik *diskrecione kontrole pristupa*, pri čemu se istovremeno obezbeđuju **integritet** stanja sredstava i **autentičnost** zahteva za njihovu potrošnju.

Svaki UTXO sadrži pridruženu skriptu zaključavanja (eng. *scriptPubKey*), kojom se definišu uslovi pod kojima taj izlaz može biti potrošen. Da bi određena transakcija validno potrošila UTXO, njen ulaz mora da obezbedi odgovarajuće podatke u okviru skripte otključavanja (eng. *script-Sig*), tako da kombinacija ove dve skripte, nakon izvršavanja nad stekom, rezultuje nenultom bajt-vrednošću (najčešće 0x01), što označava uspešnu validaciju. Potrošnja UTXO zapisa smatra se nevalidnom ukoliko tokom izvršavanja dođe do prekida skripte, ukoliko stek ostane prazan ili ukoliko se na vrhu steka nalazi negativna nula (0x80), koja se u Bitcoin skriptnoj semantici tretira kao netačna vrednost.

Iako Bitcoin Skript omogućava definisanje proizvoljno složenih uslova korišćenjem dostupnog skupa instrukcija, u praksi se najčešće koristi obrazac *plaćanja na heš javnog ključa* (eng. *Pay-to-*

Public-Key-Hash, P2PKH) [163]. U ovom slučaju, pošiljalac prilaže digitalni potpis i odgovarajući javni ključ, dok skriptni mehanizam proverava da li heš dostavljenog javnog ključa odgovara vrednosti zaključanoj u referenciranom UTXO zapisu.



Slika 4.9: Prikaz validacije P2PKH skripte izvršavanjem nad stekom.

4.1.3.2 Pametni ugovori

Posmatrano iz apstraktnog ugla, svaki UTXO zapis, zbog svog jedinstvenog identifikatora i determinističkog mehanizma validacije, može se interpretirati kao instanca jednostavnog programa koji se može izvršiti tačno jednom. Ovaj program poseduje jednu implicitnu funkciju čije telo čini skripta zaključavanja *scriptPubKey*, dok se ulazni parametri prosleđuju putem skripte otključavanja *scriptSig*. Nakon uspešnog izvršavanja, dati UTXO prestaje da postoji i ne može biti ponovo iskorišćen, čime se obezbeđuje njegova jednokratna potrošivost.

Na taj način, UTXO model realizuje mehanizam izvršavanja predikata potrošnje, koji se može formalizovati sledećim algoritmom:

Algoritam 4.2 UTXO kao predikat potrošnje

```

1: function VALIDATEUTXO(scriptSig)
2:   execute scriptPubKey with input scriptSig
3:   if stack is non-empty and top(stack) = true then
4:     return true
5:   else
6:     return false
7:   end if
8: end function

```

Bitcoin sistem na ovaj način omogućava izvršavanje jednostavnih, determinističkih programa u decentralizovanom okruženju, bez potrebe za pouzdanim posrednicima, pri čemu protokol garantuje da se svaki takav program može uspešno izvršiti najviše jednom. Ovakav koncept samostalno izvršavajućih ugovornih uslova prvi je formalno definisao Nick Szabo 1994. godine [164].

Definicija 4.11. *Pametni ugovori predstavljaju programibilne ugovore u kojima su uslovi sporazuma eksplicitno definisani u obliku koda, koji se automatski izvršava kada su ispunjeni unapred određeni uslovi.*

Jedan od ključnih tehnoloških iskoraka u razvoju blokčejn sistema predstavlja uvođenje Tjuring-kompletne virtuelne mašine na nivou cele mreže, čime je omogućeno izvršavanje pametnih ugovora opšte namene. Ovi ugovori tokom svog izvršavanja mogu međusobno komunicirati i pozivati druge pametne ugovore, u potpuno decentralizovanom okruženju, čime se omogućava realizacija decentralizovanih aplikacija (eng. *Decentralized Application, dApp*).

U takvom okruženju, adrese mogu predstavljati kako korisničke naloge, tako i pametne ugovore, dok transakcije, pored prenosa vrednosti, mogu sadržati i poziv funkcije sa pripadajućim skupom parametara koji se prosleđuju pametnom ugovoru radi njegovog izvršavanja. Budući da je stanje blokčejna logički nepromenljivo, kombinacija pametnih ugovora i distribuirane knjige podataka omogućava izgradnju pouzdanog, transparentnog i verifikabilnog izvršnog okruženja.

Virtuelna mašina nije strogi preduslov za realizaciju pametnih ugovora, ali značajno pojednostavljuje njihovu implementaciju i omogućava postojanje jedinstvenog izvršnog modela na nivou cele mreže, čime se blokčejn može posmatrati kao oblik **globalnog distribuiranog računara**.

S obzirom na to da virtuelna mašina izvršava skup niskonivojskih (eng. *low-level*) instrukcija, u praksi se koriste viši programski jezici za pisanje pametnih ugovora, koji se potom prevode u odgovarajući bajtkod (eng. *bytecode*). Tokom procesa prevođenja generišu se i programski interfejsi koji olakšavaju interakciju korisnika i drugih ugovora sa datim pametnim ugovorom.

4.1.3.3 Apstrakcija naloga

Jedan od ključnih izazova za šire usvajanje blokčejn tehnologija predstavlja složenost korisničkog iskustva, naročito u pogledu upravljanja privatnim ključevima i semenskim frazama. Ovakav model autentifikacije zasnovan na kriptografiji prenosi punu odgovornost na korisnika za bezbedno čuvanje poverljivih podataka, pri čemu njihov gubitak ili kompromitacija može dovesti do nepovratnog gubitka sredstava. Dodatno, model naplate transakcionih naknada predstavlja dodatni sloj složenosti za krajnje korisnike.

S obzirom na to da većina korisnika ne poseduje napredno tehničko znanje, neophodno je da decentralizovane aplikacije apstrahuju ove složene mehanizme kako ne bi predstavljali prepreku njihovoj upotrebi. U tom kontekstu, apstrakcija naloga (eng. *account abstraction*) predstavlja pristup koji ima za cilj unapređenje interakcije između korisnika i blokčejn sistema, omogućavajući fleksibilnije modele autentifikacije i upravljanja nalogima [165].

Za razliku od tradicionalnog modela u kojem je nalog direktno vezan za jedan privatni ključ, apstrakcija naloga omogućava implementaciju programabilnih naloga u vidu pametnih ugovora, čime se logika autorizacije i validacije transakcija može prilagoditi konkretnim potrebama korisnika. Na taj način, korisnici mogu definisati različite mehanizme potpisivanja transakcija, uključujući višestruke potpise, socijalni oporavak naloga, kao i integraciju sa drugim sistemima autentifikacije.

Dodatno, ovakav pristup omogućava fleksibilnije modele plaćanja transakcionih naknada, uključujući scenarije u kojima treće strane (npr. same aplikacije) preuzimaju troškove izvršavanja transakcija, kao i korišćenje alternativnih tokena umesto primarne kriptovalute mreže.

U literaturi i praksi razlikuju se dva osnovna pristupa implementaciji apstrakcije naloga:

1. **Apstrakcija naloga zasnovana na pametnim ugovorima** [166] - pristup koji omogućava implementaciju apstrakcije naloga bez izmene osnovnog protokola, pri čemu korisnici implementiraju sopstvene naloge kao pametne ugovore. U ovom modelu uvodi se alternativni mehanizam obrade transakcija (npr. *UserOperation* strukture i alternativni skupovi nepotvrđenih transakcija), pri čemu se zahtevi agregiraju i izvršavaju putem specijalizovanog pametnog

ugovora na mreži.

2. **Urođena apstrakcija naloga** - pristup u kojem je apstrakcija naloga integrisana direktno u protokol blokčejna, čime se eliminiše potreba za dodatnim slojevima i omogućava urođena podrška za fleksibilne modele autentifikacije i izvršavanja transakcija.

Implementacija ovog pristupa i dalje predstavlja aktivnu oblast istraživanja, sa značajnim potencijalom da unapredi upotrebljivost blokčejn sistema i omogući njihovo šire usvajanje.

4.2 Upravljanje blokčejnom

U tradicionalnim organizacionim sistemima, odluke o promenama i razvoju sistema donosi ograničena i jasno definisana grupa ljudi, čime se upravljanje ostvaruje kroz centralizovanu ili hijerarhijski strukturisanu kontrolu. Nasuprot tome, jedan od osnovnih principa javnih blokčejn sistema jeste visok stepen kontrolne decentralizacije, pri čemu u proces odlučivanja mogu biti uključeni korisnici mreže, programeri pametnih ugovora, operatori čvorova, istraživači i drugi relevantni učesnici.

Upravljanje blokčejnom [167] (eng. *blockchain governance*) funkcioniše bez centralnog upravnog tela i obuhvata skup procesa, pravila i društveno-tehničkih mehanizama putem kojih zajednica definiše način predlaganja, razmatranja, usvajanja i implementacije promena protokola. Ovi procesi određuju ko ima pravo da predloži izmene, ko može učestvovati u donošenju odluka, na koji način se predlozi evaluiraju i prihvataju, kao i kako se razrešavaju neslaganja koja mogu dovesti do račvanja mreže.

Kako su blokčejn sistemi sazrevali, u praksi su se izdvojila dva dominantna modela upravljanja: upravljanje van protokola (eng. *off-chain governance*) i upravljanje unutar protokola (eng. *on-chain governance*), koji se razlikuju po načinu formalizacije odlučivanja i stepenu automatizacije procesa donošenja odluka.

Upravljanje van protokola odvija se kroz neformalne i poluformalne mehanizme, kao što su diskusije na forumima, društvenim mrežama i razvojnim platformama. U ovom modelu, pravo učešća nije ograničeno isključivo vlasništvom nad sredstvima, već se zasniva i na reputaciji, stručnosti i doprinosu zajednici, čime se podstiče šira i kvalitetnija razmena mišljenja. Ovakav pristup smanjuje rizik od brzopletih odluka i manipulacije putem finansijske moći, ali istovremeno može dovesti do dugotrajnih i slabo strukturisanih rasprava, kao i do ograničenog učešća netehničkih korisnika. Dodatni izazov predstavlja činjenica da pojedini učesnici u slučaju manjih nadogradnji mogu namerno odlagati ažuriranje softvera ukoliko procene da im predložene promene nisu u sopstvenom interesu.

Upravljanje unutar protokola sprovodi se direktno na blokčejnu, pri čemu učesnici sa odgovarajućim pravima glasa koriste pametne ugovore za formalno donošenje odluka. Ovakav pristup obezbeđuje visok stepen transparentnosti, brzu verifikaciju rezultata i automatsku primenu usvojenih promena na svim čvorovima mreže. Međutim, ovaj model je podložan problemima niske izlaznosti učesnika, kao i riziku koncentracije moći, jer mali broj adresa sa velikim udelom sredstava može imati dominantan uticaj na konačne odluke.

Zbog navedenih ograničenja, savremeni blokčejn sistemi sve češće primenjuju hibridni model upravljanja, u okviru koga se ideje i predlozi najpre razmatraju kroz vanprotokolne mehanizme, dok se konačna odluka formalizuje i izvršava putem mehanizama upravljanja unutar protokola.

Nakon postizanja saglasnosti o sadržaju predložene izmene, nova verzija protokola se najpre implementira i testira na testnoj mreži (eng. *testnet*), gde učesnici proveravaju njenu ispravnost, bezbednost i kompatibilnost. Ukoliko se testiranje završi uspešno, zajednica se usaglašava oko tačno definisanog bloka od kojeg nova pravila stupaju na snagu na glavnoj mreži.

U zavisnosti od obima i prirode promena, razlikuju se dve osnovne kategorije nadogradnji protokola:

1. **Manje promene (eng. *soft fork*)** uvode ograničene modifikacije koje ne narušavaju fundamentalnu strukturu protokola. Ove izmene su unazad kompatibilne (eng. *backward compatible*), što znači da čvorovi koji ne ažuriraju softver mogu i dalje učestvovati u mreži, ali uz poštovanje postojećih pravila.
2. **Glavne promene (eng. *hard fork*)** uvode nove funkcionalnosti ili promene koje nisu kompatibilne sa prethodnim verzijama protokola. U tom slučaju dolazi do trajnog račvanja mreže, pri čemu svaki čvor mora ažurirati softver kako bi nastavio da učestvuje u novoj verziji sistema.

4.2.1 Decentralizovane autonomne organizacije

Decentralizovane autonomne organizacije [168] (eng. *Decentralized Autonomous Organizations, DAO*) predstavljaju prirodnu evoluciju modela upravljanja unutar protokola, zasnovanu na upotrebi pametnih ugovora sa ciljem formalizacije i institucionalizacije procesa odlučivanja u blokčejn okruženju, u kome programski kod preuzima ulogu osnovnog regulatornog mehanizma, u skladu sa principom „kod je zakon” (eng. *code is law*).

U ovakvim sistemima, pravila upravljanja, raspodele resursa i izvršavanja odluka ugrađena su direktno u protokol, čime se obezbeđuje visok stepen automatizacije, transparentnosti i otpornosti na proizvoljne intervencije.

DAO modeli imaju najveći praktični i konceptualni smisao u okviru *javnih blokčejn mreža*, gde su transparentnost, otvoren pristup i verifikabilnost osnovni preduslovi njihove funkcionalnosti, zbog čega su i danas najzastupljeniji upravo u ovom okruženju.

U perspektivi daljeg razvoja, DAO sistemi se sve više posmatraju kao autonomni digitalni entiteti sposobni da koordiniraju ljudske i softverske aktere, pružaju usluge, upravljaju finansijskim tokovima i ostvaruju ekonomske ciljeve u okviru decentralizovanih tržišnih okruženja.

Ovakav razvojni pravac otvara prostor za primenu DAO modela u oblastima *elektronske trgovine*, upravljanja digitalnim platformama i automatizovanih poslovnih ekosistema, u kojima decentralizovane organizacije mogu delovati kao samostalni učesnici tržišta, u interakciji sa drugim autonomnim softverskim sistemima i ljudskim akterima [169].

4.3 Blokčejn trilema

U distribuiranim sistemima koji se oslanjaju na mrežnu komunikaciju, pojava mrežnih particija predstavlja neizbežnu realnost, usled čega Teorema 3.1 uvodi fundamentalno ograničenje nad skupom osobina koje takvi sistemi mogu istovremeno garantovati. Budući da je blokčejn po svojoj prirodi distribuirani sistem, navedeno ograničenje se neposredno primenjuje i na blokčejn mreže.

Kako ostvarivanje kontrolne decentralizacije u blokčejn sistemima zahteva odsustvo centralnog autoriteta, koordinacija i održavanje konzistentnog globalnog stanja podrazumevaju intenzivnu razmenu poruka između čvorova, čiji se obim povećava sa rastom broja učesnika mreže. U skladu sa

Teoremom 3.3, ovaj porast dovodi do izraženog režijskog opterećenja usled sinhronizacije i kontencije, čime se nameće direktan kompromis između stepena decentralizacije i sistemskih performansi.

Istovremeno, uklanjanje centralizovanog autoriteta uvodi potrebu za tolerancijom vizantijskog ponašanja učesnika. Kako formalizuje Teorema 2.2, korektno funkcionisanje decentralizovanog sistema moguće je garantovati isključivo ukoliko udeo sistemskih resursa pod kontrolom vizantijskih učesnika ostane ispod određene gornje granice. Održavanje ove pretpostavke u praksi zahteva da većinski deo relevantnih oskudnih resursa pripada velikom broju ispravnih učesnika, što dodatno povećava troškove režijskog opterećenja.

Kombinacija navedenih ograničenja dovodi do fundamentalnog kompromisa između **decentralizacije, sigurnosti i skalabilnosti**, poznatog kao *blokčejn trilema* [170]. Naime, sistem koji teži visokom stepenu kontrolne decentralizacije i snažnim sigurnosnim garancijama nužno trpi smanjenje efikasnosti i skalabilnosti, dok povećanje performansi zahteva kompromis u pogledu barem jedne od preostalih osobina. Ova trilema ne predstavlja implementacioni nedostatak konkretnih protokola, već je duboko ukorenjena u samim teorijskim osnovama blokčejn sistema.

Kod dozvolom ograničenih blokčejn sistema, pristup mreži je strogo kontrolisan, a broj učesnika unapred ograničen, zbog čega je stepen decentralizacije značajno smanjen. Ovakav model omogućava postizanje višeg stepena efikasnosti uz snažne sigurnosne garancije zasnovane na poznatom skupu učesnika. Međutim, on zahteva da organizacije samostalno uspostave, upravljaju i održavaju kompletnu infrastrukturnu osnovu sistema, bilo kroz sopstvene resurse ili korišćenjem usluga računarstva u oblaku, što podrazumeva značajne operativne i finansijske troškove, kako je razmatrano u odeljku 2.4.1.

Nasuprot tome, *javni blokčejn sistemi* su dozvolom neograničeni i oslanjaju se na već uspostavljenu globalnu infrastrukturu koju održava veliki broj nezavisnih učesnika, čime se eliminišu troškovi održavanja sopstvene infrastrukture. Međutim, *troškovi korišćenja mreže prenose se na korisnike kroz mehanizme transakcionih naknada*. Istovremeno, izražena decentralizacija ovih sistema unapređuje otpornost i sigurnost podataka na više nivoa, obuhvatajući kako infrastrukturni i aplikativni sloj, tako i krajnje korisnike.

4.3.1 Decentralizovanost blokčejn sistema

Blokčejn sistemi u svom izvornom obliku zasnovani su na Nakamoto konsenzusu, čija je osnovna ideja raspodela kontrole nad mrežom na veliki broj dozvolom neograničenih i međusobno nezavisnih učesnika. U takvom modelu, nijedan pojedinačni entitet ne poseduje odlučujuću moć nad funkcionisanjem sistema, budući da svaki učesnik može konkurisati za proizvodnju narednog bloka. Izbor kanonske verzije lanca ne vrši se putem eksplicitnog glasanja, već implicitno kroz tržišni mehanizam konkurencije konsenzusnih resursa, čime se kao važeći prihvata lanac sa najvećom akumuliranom težinom tog resursa, u skladu sa protokolskim pravilom izbora lanca.

Nasuprot tome, blokčejn sistemi zasnovani na BFT konsenzus mehanizmima koriste unapred definisan skup validatora, pri čemu se u svakom krugu konsenzusa jedan čvor formalno bira od strane protokola kao predlagач bloka, dok ostali validatori učestvuju u procesu eksplicitnog glasanja. Ovakav pristup podrazumeva razmenu poruka čiji broj uobičajeno raste kvadratno sa brojem validatora, što *praktično ograničava veličinu validacionog skupa*. Zbog toga se BFT sistemi projektuju sa relativno malim brojem validatora, čime se postiže visoka efikasnost i deterministička finalnost, ali se istovremeno umanjuje stepen kontrolne decentralizacije.

Međutim, kvantitativno merenje kontrolne decentralizacije u realnim sistemima predstavlja izu-

zeto složen problem, koji i dalje ostaje jedno od otvorenih istraživačkih pitanja savremene literature. Postojeći radovi [171] ukazuju na to da ne postoji univerzalno prihvaćen indeks koji bi na jedinstven način kvantifikovao stepen decentralizacije, te se analiza decentralizacije sprovodi kroz posmatranje pojedinačnih slojeva blokčejn sistema i njihovog relativnog doprinosa ukupnoj strukturi kontrole.

Savremeni pristupi analizi decentralizacije [172] započinju izborom posmatranog sloja sistema, pri čemu se posebno izdvajaju dva osnovna sloja zajednička svim blokčejn sistemima, čije se karakteristike mogu direktno izvesti iz podataka zapisanih u samom blokčejnu:

1. **Sloj konsenzusa** (eng. *consensus layer*) obuhvata direktno učešće čvorova u konsenzusnom protokolu kroz kreiranje i validaciju blokova koji proširuju globalnu knjigu evidencije, pri čemu ovaj sloj ima neposredne implikacije po bezbednost sistema,
2. **Sloj tokenomije** (eng. *tokenomics layer*) obuhvata raspodelu ekonomske vrednosti unutar sistema i povezan je sa mehanizmima upravljanja i korisnošću mreže, pri čemu kritične odluke često zavise od strukture vlasništva nad tokenima.

Radi procene decentralizacije izabranog sloja, neophodno je najpre identifikovati ključni resurs tog sloja i analizirati njegovu raspodelu među učesnicima. Pre same evaluacije decentralizacije sprovodi se faza *predobrade prikupljenih podataka*, koja obuhvata grupisanje adresa koje pripadaju istom realnom entitetu, izbor perioda posmatranja za procenu udela resursa, identifikaciju aktivne populacije učesnika, definisanje učestalosti uzorkovanja podataka, kao i određivanje pragova za uključivanje ili isključivanje entiteta sa zanemarljivim udelima resursa.

Kako ne postoji jedinstven indeks koji bi objedinio sve metrike u jednu numeričku vrednost, nakon sprovedene predobrade prikupljeni podaci se podvrgavaju kvantitativnoj analizi primenom više komplementarnih metrika decentralizacije, među kojima se najčešće koriste:

1. **Šenonova entropija** [173], koja kvantifikuje stepen neizvesnosti u predviđanju proizvođača narednog bloka, pri čemu veće vrednosti ukazuju na ravnomerniju raspodelu moći,
2. **Ginijev koeficijent** [174], koji meri neravnomernost raspodele resursa među učesnicima, gde veće vrednosti označavaju viši stepen koncentracije,
3. **Nakamoto koeficijent**, koji predstavlja minimalan broj entiteta čija bi koordinacija omogućila kontrolu većine resursa u mreži,
4. **τ indeks**, koji generalizuje Nakamoto koeficijent određivanjem broja entiteta potrebnih za kontrolu unapred definisanog udela τ ukupnih resursa,
5. **Koncentracioni odnos** [175], koji meri koliki deo ukupnih resursa pripada skupu od n najmoćnijih učesnika,
6. **Herfindahl-Hiršmanov indeks** [176], koji kvantifikuje ukupni stepen koncentracije resursa u sistemu i potiče iz oblasti antimonopolskih analiza.

Javne blokčejn mreže karakteriše visok stepen decentralizacije, koji proizlazi iz njihove otvorene i transparentne prirode. Većina ovih sistema razvija se kao softver otvorenog koda (eng. *open-source*), čime se omogućava javna proverljivost implementiranih algoritama i smanjuje rizik od netransparentne algoritamske kontrole. Ipak, treba napomenuti da u pojedinim blokčejn sistemima ove vrste određeni delovi šireg ekosistema (npr. klijentske implementacije, infrastruktura ili aplikacioni sloj) mogu ostati zatvoreni, što uvodi potencijalne tačke centralizacije.

S obzirom na to da su javne blokčejn mreže dozvolom neograničene, mehanizmi cenzure i ograničavanja pristupa znatno su otežani, budući da ne postoji centralni autoritet koji bi mogao selektivno blokirati transakcije. Međutim, u praksi se uočava koncentracija moći kod velikih provajdera infrastrukture u centrima podataka (eng. *data centers*) [177], što može dovesti do ograničenih oblika **arhitektonske centralizovanosti** na nivou pristupa mreži.

Uprkos tome, otvorena priroda sistema omogućava korisnicima da pokretanjem sopstvenih čvorova na ličnim računarima ili u *bare-metal* okruženjima nezavisno verifikuju stanje mreže i šalju transakcije, čime se dodatno doprinosi smanjenju centralizacije i povećanju otpornosti sistema.

Podaci jednom zapisani na blokčejnu ostaju trajno dostupni, što predstavlja ograničenje u pogledu zaštite ličnih podataka i ne eliminiše u potpunosti obrasce nadzornog kapitalizma. Iako se identiteti korisnika predstavljaju pseudonimno putem kriptografskih adresa, povezivanje tih adresa sa identitetima iz stvarnog sveta može dovesti do potencijalne zloupotrebe podataka, budući da su sve transakcije javno dostupne i trajno zabeležene [178].

4.3.2 Sigurnost blokčejn sistema

Sigurnost predstavlja fundamentalno svojstvo blokčejn sistema, budući da obezbeđuje jedinstvenost, neizmenljivost i konzistentnost distribuirane knjige evidencije, čak i u prisustvu vizantijskih čvorova. Očuvanje ovih svojstava mora biti garantovano bez izuzetaka, jer kompromitovanje bilo kog od njih dovodi u pitanje osnovnu funkcionalnost i smisao postojanja blokčejn sistema.

Usled asinhronne prirode distribuiranih mreža, blokovi se ne propagiraju istovremeno ka svim čvorovima, što može dovesti do privremenog postojanja konkurentnih verzija lanca, odnosno do pojave račvanja.

Blokčejn sistemi zasnovani na BFT konsenzus mehanizmima postižu saglasnost kroz razmenu poruka i glasanje između učesnika mreže. U većini implementacija proces konsenzusa organizovan je kroz više uzastopnih faza ili rundi, pri čemu jedan od čvorova privremeno preuzima ulogu glavnog čvora i predlaže narednu vrednost ili blok [179]. Ostali učesnici zatim proveravaju predlog, razmenjuju poruke i glasaju o njegovom prihvatanju u skladu sa pravilima konkretnog protokola.

Ukoliko se tokom izvršavanja protokola postigne potreban kvorum glasova, predložena vrednost smatra se prihvaćenom. U suprotnom, pokreće se procedura promene pogleda (eng. *view change*) ili nova runda glasanja, nakon čega se proces konsenzusa nastavlja sa novim predlogom ili novim glavnim čvorom. Na taj način sistem može da očuva saglasnost čak i u prisustvu zlonamernih ili neispravnih učesnika.

Ovakav pristup obezbeđuje determinističku finalnost, zbog čega BFT sistemi pružaju visok nivo bezbednosti i konzistentnosti stanja. Dodatno, radi prilagođavanja blokčejn okruženju i ostvarivanja boljih performansi, razvijene su modernije varijante BFT protokola [180].

Nasuprot tome, Nakamoto konsenzus dozvoljava privremeno postojanje konkurentnih verzija lanca, koje same po sebi ne predstavljaju narušavanje sigurnosti, budući da protokol garantuje njihovo naknadno razrešenje i konvergenciju ka jedinstvenoj globalnoj istoriji transakcija. Međutim, ukoliko napadač raspolaže dovoljnim udelom resursa relevantnih za proces konsenzusa, može proizvesti alternativnu istoriju transakcija koja biva prihvaćena kao validna od većine čvorova. Time postaje moguće izvršiti reorganizaciju lanca, poništiti prethodno potvrđene transakcije, ostvariti dvostruku potrošnju i narušiti neizmenljivost evidencije, čime se sigurnost sistema direktno kompromituje.

Iz navedenog sledi da sigurnost blokčejn sistema zavisi od sposobnosti konsenzusnog mehanizma da obezbedi dugoročnu konvergenciju ka jedinstvenoj verziji lanca, čak i u prisustvu zlonamernih

učesnika.

U tom kontekstu, stopa račvanja lanca F iako ne predstavlja potpunu meru sigurnosti često se koristi kao indikativna metrika stabilnosti konsenzusa, pri čemu se u pojedinim radovima [181] njen inverz koristi kao aproksimativna mera sigurnosti sistema:

$$Sigurnost = \frac{1}{F}$$

4.3.3 Skalabilnost blokčejn sistema

U skladu sa Definicijom 3.5, skalabilnost blokčejn sistema odnosi se na sposobnost mreže da, uz povećanje raspoloživih resursa, ostvari porast efektivnog kapaciteta obrade transakcija, pri čemu se istovremeno očuvaju prihvatljive performansne karakteristike i bezbednosna svojstva sistema.

Uprkos intenzivnom razvoju ove tehnologije, skalabilnost i dalje predstavlja jedno od ključnih otvorenih istraživačkih pitanja, budući da se upravo ograničenja u pogledu kapaciteta i performansi često identifikuju kao glavna prepreka širem usvajanju blokčejn tehnologije u savremenim informacionim sistemima. Sistematična analiza izazova skalabilnosti u blokčejn mrežama [182] ukazuje da na skalabilnost utiče skup međuzavisnih faktora, uključujući propusnost transakcija, broj čvorova u mreži, veličinu distribuirane evidencije, veličinu bloka, troškove i kašnjenja komunikacije između čvorova, vreme finalizacije transakcija, kao i trošak izvršavanja transakcija.

Navedeni faktori su snažno međusobno povezani i u velikoj meri *određeni svojstvima primenjenog konsenzusnog mehanizma*, budući da upravo konsenzus definiše ključne protokolske parametre, kao što su veličina bloka i vremenski interval između uzastopnih blokova. Savremeni konsenzusni mehanizmi, u svom osnovnom obliku, još uvek ne uspevaju da obezbede nivo propusnosti i latencije neophodan za širok spektar industrijskih primena, pri čemu se ograničenje dodatno izražava činjenicom da se efektivna vrednost TPS ne povećava sa rastom broja čvorova.

Propusnost transakcija se najčešće izdvađa kao centralna metrika u analizi skalabilnosti blokčejn sistema, budući da kvantitativno odražava sposobnost sistema da obradi zahteve u jedinici vremena. Iako propusnost ne predstavlja samu skalabilnost, već njenu operativnu meru, ona se u praksi najčešće koristi kao njen neposredni pokazatelj.

$$Skalabilnost = \frac{n}{T}$$

U savremenim javnim blokčejn sistemima osnovni sloj (L1) projektovan je sa primarnim ciljem očuvanja visokog stepena decentralizacije i bezbednosti. Ovaj sloj obezbeđuje autoritativnu verziju globalnog stanja sistema i predstavlja krajnju instancu verifikacije i konsenzusne finalnosti.

Paralelno sa tim, razvijaju se zasebni blokčejn sistemi koji predstavljaju više slojeve (L2), a koji funkcionišu izvan osnovnog lanca i imaju za cilj povećanje efektivne propusnosti sistema [146]. Ovi sistemi često primenjuju različite modele izvršavanja i konsenzusa u odnosu na osnovni sloj, te su u praksi često centralizovaniji. Njihova primarna uloga ogleda se u obradi transakcija i privremenom održavanju stanja, dok se agregirani rezultati periodično zapisuju na osnovni sloj, gde se obezbeđuje njihova kriptografska verifikacija i konačna sigurnost.

Na ovaj način se postiže povećanje skalabilnosti sistema, budući da se osnovni sloj rasterećuje od neposredne obrade svih transakcija, dok se istovremeno zadržavaju njegova bezbednosna i decentralizovana svojstva. Slojevit arhitektura omogućava paralelizaciju obrade, smanjenje komunikacionog opterećenja i povećanje efektivne propusnosti bez narušavanja sigurnosnog modela osnovnog lanca.

Glava 5

Arhitektura Eterijum blokčejna

Eterijum [183, 184] (eng. *Ethereum*) predstavlja prvu blokčejn platformu koja je uvela Turing-kompletne pametne ugovore, čime prevazilazi ulogu distribuirane knjige evidencije i evoluirala u koncept globalnog računara (eng. *world computer*). Zbog toga se u savremenoj akademskoj literaturi i industrijskoj praksi prepoznaje kao jedna od najaktivnijih i tehnološki najrazvijenijih platformi za razvoj decentralizovanih aplikacija [185]. Empirijske analize aktivnosti programerske zajednice ukazuju na visok intenzitet razvoja, koji se ogleda u velikom broju izmena (eng. *commit*) u ključnim repozitorijumima u okviru Eterijum ekosistema. Ovi pokazatelji potvrđuju centralnu ulogu Eterijuma u procesu tehnoloških inovacija unutar šireg blokčejn prostora [186].

Počevši od 15. septembra 2022. godine, kada je izvršena nadogradnja *Merge*, mreža Eterijuma koristi Nakamoto konsenzus zasnovan na dokazivanju uloga (eng. *Proof of Stake, PoS*), u okviru kojeg se kao oskudni resurs zaključava primarna kriptovaluta, pri čemu učesnici postaju validatori koji mogu biti nagrađeni ili kažnjeni u zavisnosti od svog ponašanja. Prema dostupnim podacima, od drugog kvartala 2024. godine mreža obuhvata približno jedan milion aktivnih validatora, uz umerene fluktuacije tog broja tokom vremena [187]. Takođe, ukupna ponuda u opticaju iznosi približno 120 miliona jedinica nakon pomenute nadogradnje, od čega je približno 38 miliona jedinica zaključano u svrhu održavanja mreže.

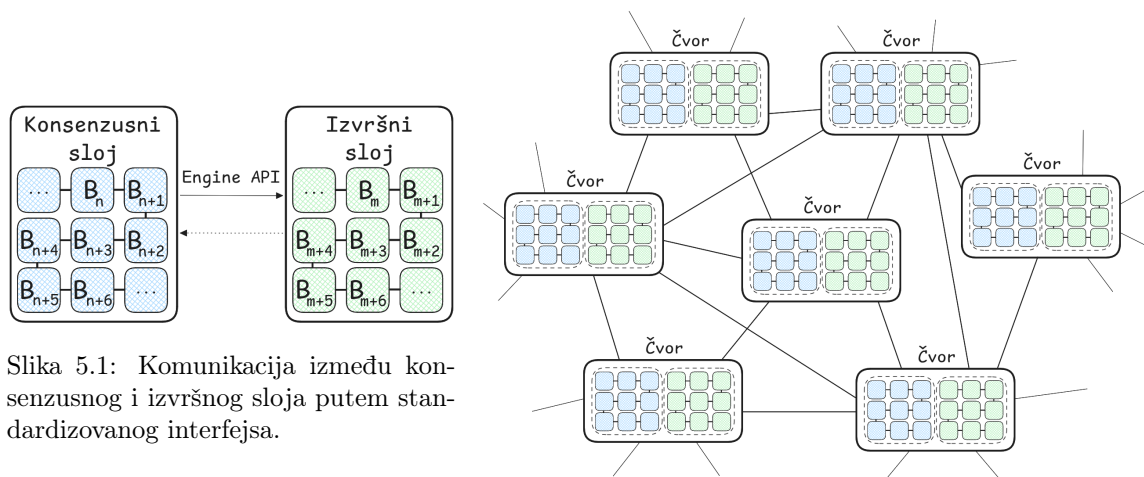
Analiza ponašanja validatora u dužem vremenskom periodu [188] ukazuje na izuzetno visok stepen učešća u konsenzusnom procesu, sa prosečnim nivoom participacije od oko 99%. Povremeni padovi u učešću uglavnom su povezani sa planiranim nadogradnjama protokola ili tehničkim problemima u softverskim klijentima koje validatori koriste. Pored toga, ista studija identifikuje **visok nivo decentralizacije** konsenzusnog sloja mreže, kvantifikovan korišćenjem standardnih metrika kao što su Nakamotov koeficijent, Ginijev koeficijent i Herfindahl-Hiršmanov indeks, čime se dodatno potvrđuje robusnost i otpornost Eterijum mreže na koncentraciju moći među učesnicima.

Eterijum se najčešće modeluje kao sistem sastavljen od dva protokolarno razdvojena sloja, koji funkcionišu kao međusobno koordinisane, ali logički nezavisne mreže ravnopravnih čvorova. Svaki sloj održava sopstveno lokalno stanje relevantno za svoju funkciju u okviru protokola:

1. **Konsenzusni sloj** (eng. *Consensus Layer*), poznat i kao *lanac svetionika* (eng. *Beacon Chain*), odgovoran je za bezbednost i pravilno napredovanje sistema. Ovaj sloj koristi mehanizam konsenzusa zasnovan na dokazivanju uloga radi izbora predlagača blokova, organizacije kanonske istorije lanca, kao i sprovođenja procesa glasanja i finalizacije blokova. Na taj način obezbeđuje sigurnost, otpornost na napade i stabilnost celokupne mreže.

2. **Izvršni sloj** (eng. *Execution Layer*) predstavlja funkcionalni računarski sloj sistema i zadužen je za izvršavanje transakcija, upravljanje stanjem naloga i pametnih ugovora, kao i za definisanje semantike i pravila izvršavanja u okviru Eterijum virtuelne mašine. Ovaj sloj omogućava implementaciju i korišćenje decentralizovanih aplikacija i predstavlja osnovu funkcionalnosti Eterijum ekosistema.

Svaki validator mora imati funkcionalan pristup i konsenzusnom i izvršnom sloju protokola, zbog čega se koristi slojevita arhitektura klijenata. Razdvajanje izvršnog i konsenzusnog klijenta, naročito kada se izvršavaju na različitim uređajima, može povećati napadnu površinu sistema i uvesti dodatne bezbednosne rizike u njihovu međusobnoj komunikaciji.



Slika 5.1: Komunikacija između konsenzusnog i izvršnog sloja putem standardizovanog interfejsa.

Slika 5.2: Mreža ravnopravnih čvorova u Eterijum mreži.

Komunikacija između konsenzusnog i izvršnog sloja je dvosmerna i ostvaruje se isključivo putem standardizovanog interfejsa poznatog kao *Engine API*, koji predstavlja jedini dozvoljeni mehanizam njihove koordinacije. Proces predlaganja i validacije novog bloka odvija se kroz sledeće faze:

1. Validator izabran za predlagača u okviru konsenzusnog sloja inicira izvršnom sloju proces izgradnje novog bloka pozivom *engine_forkchoiceUpdatedV2*, prosleđujući trenutni izbor kanonskog lanca i parametre za konstrukciju bloka.
2. Izvršni klijent započinje proces izgradnje izvršnog sadržaja bloka, pri čemu bira nepotvrđene transakcije iz lokalnog skupa i deterministički ih izvršava u okviru svoje Eterijum virtuelne mašine.
3. Predlagač u konsenzusnom sloju pozivom *engine_getPayloadV2* preuzima konstruisani izvršni sadržaj, nakon čega ga integriše sa konsenzusnim podacima u jedinstvenu strukturu bloka.
4. Formirani blok se kriptografski potpisuje i propagira ostalim validatorima putem konsenzusnog sloja.
5. Ostali validatori proveravaju konsenzusni deo bloka u skladu sa pravilima protokola, nakon čega izvršni sadržaj bloka prosleđuju svom izvršnom klijentu putem poziva *engine_newPayloadV2*.
6. Izvršni sloj validira blok proverom zaglavlja i determinističkim izvršavanjem svih transakcija, pri čemu vraća odgovarajući status validacije.

7. Na osnovu rezultata validacije, validatori učestvuju u procesu potvrđivanja bloka emitovanjem potvrda ukoliko je blok validan, dok se nevalidni blokovi odbacuju i ne učestvuju u daljem procesu konsenzusa.

5.1 Konsenzusni sloj Eterijum mreže

Vreme u konsenzusnom sloju podeljeno je na epohe (eng. *epochs*), pri čemu se svaka epoha sastoji od 32 slota, dok trajanje pojedinačnog slota iznosi 12 sekundi. Na osnovu toga proizlazi da trajanje jedne epohe iznosi 6 minuta i 24 sekunde.

Da bi učesnik mreže stekao status validatora, neophodno je da najpre zaključi najmanje 32 ETH [189], primarne kriptovalute Eterijum blokčejna, u ugovoru za ulaganje (eng. *staking contract*). Time učesnik prelazi u stanje validatora u redu za aktivaciju (eng. *activation queue*), gde čeka uključivanje u proces validacije. Nakon isteka perioda čekanja, validator postaje deo skupa aktivnih validatora i može učestvovati u procesu predlaganja i validacije blokova.

Razlog za čekanje jeste postojanje ograničenja broja validatora koji se mogu aktivirati tokom jedne epohe, definisanog parametrom ograničenja protoka (eng. *churn limit*) [190], u cilju kontrole dinamike ulaska i izlaska validatora [191], čime se doprinosi stabilnosti mreže.

S obzirom na to da blokčejn predstavlja deterministički distribuirani sistem, neophodno je obezbediti izvor pseudonasumičnosti za izbor predlagača bloka koji je istovremeno javno proverljiv i unapred nepredvidiv, kako bi se sprečila mogućnost uticaja napadača na raspodelu konsenzusnih uloga. U konsenzusnom sloju Eterijuma ovaj zahtev realizuje se putem distribuiranog mehanizma generisanja slučajnosti poznatog kao *RANDAO* [192].

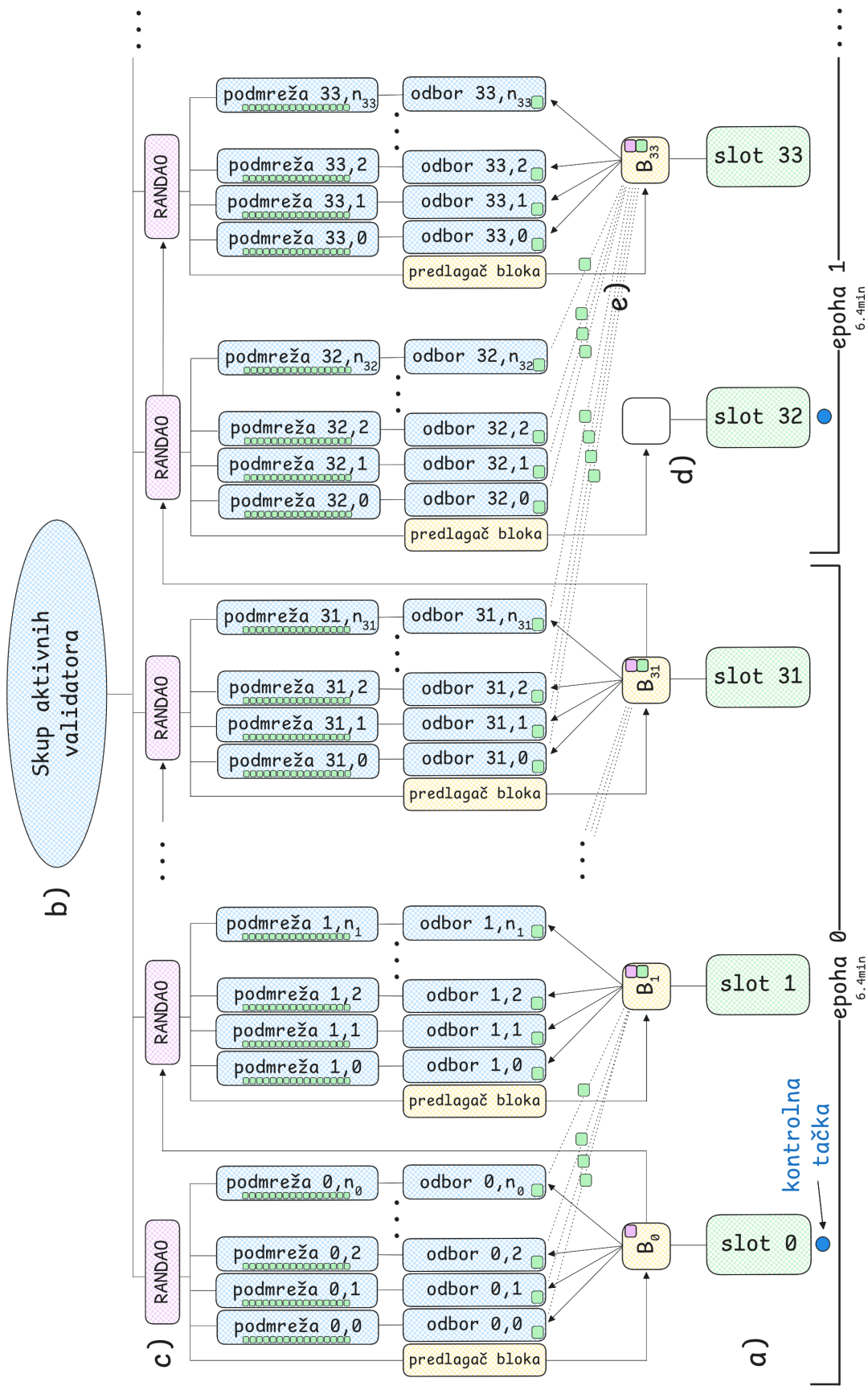
RANDAO predstavlja kolektivni kriptografski protokol u kojem validatori postepeno doprinose generisanju zajedničke pseudonasumične vrednosti. Ova vrednost ažurira se kada validator izabran za trenutni slot predloži novi blok, pri čemu je obavezan da u zaglavlje bloka uključi odgovarajuću vrednost (eng. *randao reveal*). Na osnovu tako generisane RANDAO vrednosti, na početku svake epohe vrši se deterministička, ali pseudonasumična permutacija skupa aktivnih validatora.

Tom prilikom validatori se raspoređuju u odbore (eng. *committees*) približno jednake veličine, pri čemu minimalna veličina odbora mora biti veća od 128 validatora iz bezbednosnih razloga. U okviru jednog slota može postojati najviše 64 odbora, pri čemu broj odbora zavisi od broja aktivnih validatora, dok na verovatnoću izbora validatora utiče visina njegovog efektivnog uloga.

Predlagač bloka prilikom kreiranja novog bloka, putem povezanog izvršnog klijenta, bira transakcije iz svog lokalnog skupa nepotvrđenih transakcija i uključuje ih u izvršni sadržaj bloka. Nakon interne provere izvršnog sadržaja i njegove integracije sa konsenzusnim podacima u jedinstvenu strukturu bloka, predloženi blok se distribuira ostalim validatorima u odgovarajućem odboru.

U ovoj fazi preostali validatori proveravaju validnost bloka najpre na konsenzusnom, a zatim na izvršnom sloju, i ukoliko su obe faze uspešne, glasaju za blok koji smatraju delom kanonske grane lanca emitovanjem najviše jedne atestacije (eng. *attestation*), odnosno potvrde koja sadrži sledeće podatke:

1. **slot** - broj slota na koji se atestacija odnosi,
2. **indeks** - indeks odbora kojem validator pripada u trenutnom slotu,
3. **heš bloka na vrhu lanca** (eng. *beacon block root*) - heš bloka koji validator trenutno smatra vrhom lanca,



Slika 5.3: Struktura konsenzusnog sloja u Eterijum mreži.

- Podjela vremena na slotove.
- Skup aktivnih validatora koji učestvuju u predlaganju novih blokova.
- Protokol RANDAO koji pseudonasumično raspoređuje skup aktivnih validatora na početku svakog slot.
- Slot u kojem validator nije predložio blok, usled čega nije došlo do promene RANDAO pseudonasumične vrednosti.
- Prikupljanje atestacija nastalih usled izostanka predlaganja bloka u prethodnom slotu.

4. **izvorna kontrolna tačka** (eng. *source*) - kontrolna tačka poslednje opravdane (eng. *justified*) kontrolne tačke koja je validatoru poznata,
5. **ciljna kontrolna tačka** (eng. *target*) - kontrolna tačka trenutne epohe, odnosno prvi blok trenutne epohe.

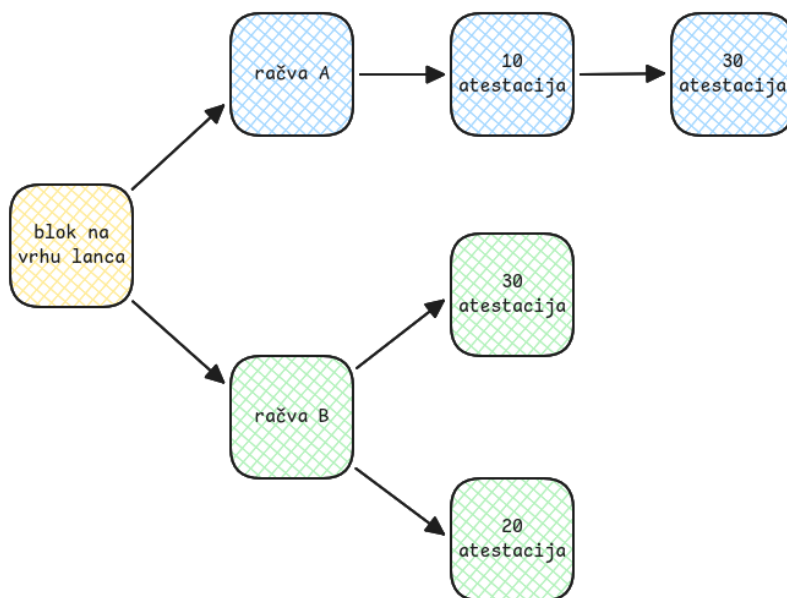
Kako se svaki odbor sastoji od velikog broja validatora, direktna propagacija pojedinačnih atestacija dovela bi do zagušenja mreže, u skladu sa Teoremom 3.3. Iz tog razloga, svakom odboru dodatno se dodeljuje komunikaciona podmreža (eng. *subnets*), pri čemu se svaki validator nalazi u okviru tačno jedne podmreže.

Unutar svake podmreže, šesnaest validatora bira se za ulogu agregatora (eng. *aggregators*), koji prikupljaju atestacije ostalih validatora iz istog odbora i kombinuju njihove kriptografske potpise u jedinstven agregatni potpis primenom BLS šeme agregacije potpisa [193] nakon čega emituju dobijeni agregatni potpis široj mreži. Ovakav mehanizam omogućava efikasnu verifikaciju velikog broja atestacija uz minimalan komunikacioni i računarski trošak.

Validator koji je izabran za predlagača bloka uključuje u predloženi blok agregirane BLS atestacije iz svake komunikacione podmreže, pri čemu se za svaku podmrežu bira agregacija sa najvećom podrškom validatora. Na taj način agregirane atestacije kombinuju se u jedinstveni BLS potpis, koji postaje sastavni deo strukture bloka.

5.1.1 Račvanja u konsenzusnom sloju

Usled asinhronne prirode distribuirane mreže, predloženi blok ne dospeva do svih validatora istovremeno. Kao posledica toga, pojedini validatori mogu primiti novi blok tek u narednom slotu. Validatori koji nisu primili novi blok nastavljaju da glasaju za poslednji blok koji im je poznat. Ukoliko novi predlagač nije upoznat sa prethodno predloženim blokom, on može konstruisati blok koji se nadovezuje direktno na poslednji poznati blok, čime dolazi do privremenog račvanja lanca.



Slika 5.4: Primer račvanja lanca, pri čemu se u prvom slučaju blokovi linearno nadovezuju na inicijalnu račvu, dok u drugom dolazi do dodatnog račvanja ogranka sa većim brojem atestacija.

S obzirom na to da svaka atestacija sadrži informaciju o bloku koji validator trenutno vidi na vrhu blokčejna, validator u svakom narednom slotu ima mogućnost da promeni svoj izbor i emituje novu atestaciju za blok na suprotnoj grani, ukoliko ne želi da glasa za istu granu. Iz tog razloga, u procesu izbora glavne grane u obzir se uzima isključivo poslednja važeća atestacija svakog validatora, odnosno atestacija sa najvećim brojem slotova koju je validator emitovao i koja je uključena u agregirane BLS potpise (eng. *Latest Message Driven, LMD*).

Kako se u Eterijum mreži zaključava primarna kriptovaluta u vidu uloga validatora, izbor grane koja će biti proglašena glavnom u slučaju račvanja zasniva se na *pravilu najtežeg lanca*, koje predstavlja varijaciju klasičnog pravila najdužeg lanca. Ovo pravilo ne posmatra broj blokova na grani, već **težina uloga** predstavlja ukupnu količinu ETH zaključanu u podstablu posmatranog bloka, koja uzima uzimajući u obzir poslednje atestacije svih validatora koji su glasali za njega ili njegove potomke (eng. *Greedy Heaviest-Observed Sub-Tree, GHOST*).

U slučaju račvanja, Nakamoto konsenzus obezbeđuje verovatnosnu finalnost Eterijum blokčejna primenom kombinovanog mehanizma *LMD-GHOST*, koji koristi poslednje atestacije validatora za deterministički izbor grane sa najvećom ukupnom težinom.

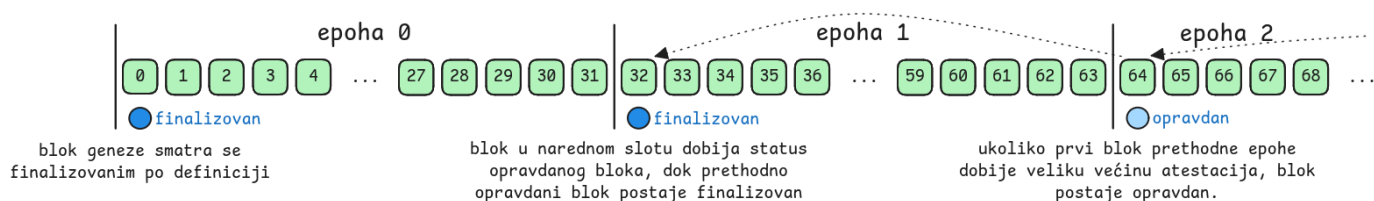
5.1.2 Mehanizam finalizacije blokova

Eterijum blokčejn može funkcionisati isključivo primenom mehanizma izbora glavne grane zasnovanog na algoritmu *LMD-GHOST*, čime se obezbeđuje verovatnosna finalnost blokova. Međutim, kako bi se obezbedila jaka, odnosno eksplicitna finalnost, Eterijum protokol koristi dodatni mehanizam konsenzusa poznat kao *Casper Friendly Finality Gadget, Casper FFG* [194]. Kombinacijom ova dva mehanizma ostvaruje se hibridni konsenzusni model, u kojem se izbor glavne grane vrši pomoću *LMD-GHOST* algoritma, dok se nepovratnost stanja sistema obezbeđuje putem Casper FFG mehanizma [195].

Casper FFG predstavlja BFT mehanizam finalizacije koji se ne bavi predlaganjem blokova, već se primenjuje nad već formiranim lancem blokova. Finalizacija se odvija na nivou epohe, pri čemu se kao kontrolne tačke (eng. *checkpoints*) definišu prvi blokovi svake epohe, kojima se dodeljuje posebna semantička uloga u procesu finalizacije.

Kako svaka atestacija sadrži informacije o izvornoj i ciljnoj kontrolnoj tački, validatori putem svojih atestacija glasaju za prelaz između dve uzastopne kontrolne tačke pa atestacijama istovremeno učestvuju i u mehanizmu izbora grane i u procesu finalizacije.

Ukoliko **velika većina** (najmanje dve trećine ukupnog uloga aktivnih validatora) u okviru jedne epohe potvrdi isti prelaz između kontrolnih tačaka, ciljane kontrolna tačka se smatra *opravdanom*. Ukoliko se u narednoj epohi opravda novi prelaz koji polazi od ove opravdane kontrolne tačke, ona postaje *finalizovana*. Na taj način, finalizacija zahteva uspešno opravdavanje u dve uzastopne epohe.



Slika 5.5: Mehanizam finalizacije blokova kroz dve uzastopno opravdane epohe.

Ovakav dvostepeni proces obezbeđuje toleranciju na vizantijsko ponašanje validatora, ali Casper

FFG algoritam ne pokušava da finalizuje svaku kontrolnu tačku redom, jer u slučaju konfliktnog glasanja dolazi da u trenutnoj rundi ne bude finalizovana kontrolna tačka. U slučajevima kad se ovo desi Eterijum mreža nastavlja sa verovatnosnom finalnošću zasnovanom na *LMD-GHOST* algoritmu.

Nakon konfliktnog glasanja kad se ne finalizuje poslednja opravdana tačka, Casper FFG algoritam nastavlja sa radom gde se ista opravdana kontrolna tačka uzima za glasanje ali i nova ciljna kontrolna tačka u trenutnoj epohi, gde će se glasati za ovaj par. Ovim ponašanjem se vidi da je finalizacija skokovita jer nakon finalizacije sve kontrolne tačke između dva finalizovane kontrolne tačke ostaju implicitno nepovratne, iako formalno nisu finalizovane.

Ukoliko u konsenzusnom sloju protekne više od četiri uzastopnih epoha bez finalizacije kontrolnih tačaka, protokol automatski ulazi u režim *curenja neaktivnosti* (eng. *inactivity leak*). Cilj ovog mehanizma je uspostavljanje uslova neophodnih za povratak konačnosti lanca, čime se obezbeđuje otpornost sistema na ovakav tip poremećaja.

Protokol curenja neaktivnosti postepeno umanjuje ulog nedostupnih validatora i validatora koji kontinuirano šalju atestacije koje ne doprinose formiranju većine, pri čemu je visina kazne jednaka nagradi koju bi validator ostvario da je atestaciju poslao, sve dok iskreni validatori ne ostvare dovoljnu težinu uloga što im omogućava da finalizuju lanac. Ovaj mehanizam istovremeno podstiče i iskrene, ali neaktivne validateore da se što pre ponovo uključe u protokol, jer se njihov ulog ekonomski umanjuje.

Funkcionalnost protokola curenja neaktivnosti uspešno je demonstrirana na testnoj mreži *Medalla* [196], gde su iskreni validatori uspeali da se ponovo usaglasе oko aktuelnog vrha lanca i povrate finalizaciju.

5.1.3 Bezbednosni mehanizmi konsenzusa

Sve dok u procesu generisanja slučajnosti tokom jedne epohe postoji makar jedan iskren validator koji pravilno doprinosi RANDAO mehanizmu, dobijena pseudonasumična vrednost ostaje nepredvidiva za potencijalnog napadača, čime se obezbeđuje nepredvidivo raspoređivanje validatora u odbore.

Radi očuvanja bezbednosnih garancija konsenzusnog protokola, Eterijum sistem teži da obezbedi da veličina svakog odbora ne padne ispod 128 validatora, čime se smanjuje verovatnoća uspešnih koordinisanih napada i pojedinačne manipulacije procesom atestacija.

Činjenica da su uloge validatora za svaki slot deterministički poznate na početku epohe omogućava primenu sofisticiranih MEV strategija [197], koje mogu negativno uticati na krajnje korisnike kroz manipulaciju redosledom i selekcijom nepotvrđenih transakcija.

Istovremeno, ova predvidivost otvara prostor za ciljane napade na predlagače blokova, poput uskraćivanja usluge kroz masovno slanje nepoželjnih poruka, sa ciljem da se predlagač privremeno učini nedostupnim ostatku mreže i time uspori napredovanje sistema. Potencijalno ublažavanje ovakvih napada razmatra se kroz koncept tajnog vođe [198], međutim efikasna realizacija ovog pristupa i dalje predstavlja otvoreno istraživačko pitanje.

Međutim, izbor trajanja epohe od 32 slotova, odnosno 6 minuta i 24 sekunde, predstavlja inženjerski kompromis kojim se balansiraju bezbednost i efikasnost sistema. Ovakvo vremensko ograničenje značajno smanjuje prostor za koordinisane napade, budući da napadač raspolaže ograničenim vremenom za identifikaciju cilja, orkestraciju napada i njegovo sprovođenje pre promene raspodele uloga u narednoj epohi.

5.1.3.1 Mehanizam sečenja zlonamernih validatora

Kako je za učešće u konsenzusnom procesu neophodno zaključati depozit od najmanje 32 ETH, **bezbednost Eterijum sistema** se u velikoj meri oslanja na pažljivo dizajnirane ekonomske podstice vezane za oskudan resurs.

Validatori ostvaruju nagrade za korektno izvršavanje protokola, pri čemu predlagač bloka dobija nagradu za uspešno uključivanje bloka u lanac, dok validatori koji šalju validne atestacije za taj blok dobijaju proporcionalno manji deo ukupne nagrade sistema.

Nasuprot tome, izostanak predlaganja bloka ili predlaganje nevalidnog bloka ne povlači direktnu finansijsku kaznu, već predstavlja propuštenu priliku za ostvarivanje odgovarajuće nagrade. Pored toga, protokol predviđa kazne za nemarno ponašanje u obliku curenja neaktivnosti, ukoliko su za to ispunjeni uslovi, kao što je prethodno objašnjeno.

Za razliku od Bitcoinovog konsenzus mehanizma zasnovanog na dokazu uloženog rada, Eterijumov konsenzus zasnovan na dokazu uloženog uloga uvodi složenije mehanizme glasanja i finalizacije, što povećava površinu potencijalnih napada, ali istovremeno omogućava preciznije i strože bezbednosne garancije.

Bezbednost se obezbeđuje kroz kazneni mehanizam poznat kao sečenje (eng. *slashing*), kojim se detektuje i sankcioniše ponašanje validatora koje ugrožava ispravnost konsenzusa. Validator koji prekrši pravila biva isključen iz skupa aktivnih validatora, a deo njegovog zaključanog depozita se trajno oduzima, čime se snažno destimuliše odstupanje od protokolom definisanih pravila.

Sečenje se primenjuje u sledećim slučajevima, koji se svode na zlonamerno predlaganje blokova ili neispravno glasanje:

1. Predlaganje više od jednog bloka za isti slot, čime validator direktno pokušava da izazove račvanje lanca, što predstavlja kršenje LMD-GHOST pravila izbora grane tzv. *LMD-GHOST double vote*,
2. Emitovanje dve atestacije takve da jedna logički obuhvata drugu u pogledu izabranih kontrolnih tačaka, čime validator pokušava da omogući retroaktivno prepisivanje istorije tzv. *FFG surround vote*,
3. Emitovanje više atestacija za različite ciljne kontrolne tačke u istoj epohi koje se nalaze u različitim granama lanca, čime se narušava jedinstvenost finalizacije tzv. *FFG double vote*.

U slučaju bilo kog od navedenih prekršaja, validator biva sankcionisan sečenjem, pri čemu se inicijalna kazna uvećava proporcionalno ukupnom broju validatora koji su sasečeni u istom vremenskom periodu. Ovakav mehanizam je dizajniran da učini koordinisane napade ekonomski neisplativim, jer u ekstremnim slučajevima može dovesti do potpunog gubitka uloženog depozita svih učesnika u napadu.

5.1.3.2 Napadi sa malom težinom uloga

Kako se stopa račvanja lanca koristi kao indikativna metrika stabilnosti konsenzusnog procesa, napadač može namerno izazivati česta račvanja sa ciljem destabilizacije mreže. Ovakvi napadi spadaju u domenu teorije igara primenjene na blokčejn sisteme [199].

Napadač može namerno zanemariti aktuelni blok i emitovati atestacije za prethodni blok, ili, u slučaju uloge predlagača, zadržavati predložene blokove i atestacije pre njihovog objavljivanja u cilju izazivanja račvanja lanca [200].

U visoko idealizovanom modelu mreže, u kojem napadač poseduje izuzetno preciznu kontrolu nad vremenom isporuke poruka, moguće je realizovati sledeće strategije:

1. **Balansirajući napad** (eng. *balancing attack*) - napadač zadržava predloženi blok i atestacije, a zatim ih objavljuje u pažljivo odabranom trenutku kako bi podelio iskrene validatore u dve disjunktne grupe koje prihvataju različite grane lanca.
2. **Odbijajući napad** (eng. *bouncing attack*) - napadač naizmenično atestira opravdane blokove na različitim granama lanca, sa ciljem sprečavanja njihove finalizacije.
3. **Napad lavine** (eng. *avalanche attack*) - u slučaju kontrole više uzastopnih predlagača, napadač akumulira zadržane blokove sve dok alternativna grana ne dostigne istu težinu kao iskrena grana lanca.

Odbrambeni mehanizmi ugrađeni u protokol daju dodatne težine porukama u trenutnom slotu ukoliko su primljene bez zakašnjenja, čime se značajno otežava realizacija prva dva napada, dok ovaj pristup ne pruža potpunu zaštitu od napada lavine [201]. Ipak, autori pokazuju da je ovaj napad izvodljiv isključivo u teorijskim, visoko idealizovanim uslovima.

Sve navedeno ukazuje da su napadi sa malom težinom uloga u praksi teško izvodljivi, jer zahtevaju nerealistične pretpostavke o mrežnim uslovima i preciznoj koordinaciji napadača. Iz perspektive napadača, efikasnija strategija ostaje akumulacija većeg uloga i primena istih napada u znatno povoljnijem režimu.

5.1.3.3 Napadi sa velikom težinom uloga

Zlonamerni napadač može nastojati da akumulira značajnu količinu uložених sredstava, čime se povećava njegov uticaj na ponašanje konsenzusnog protokola. U nastavku se razmatraju referentni pragovi udela ukupnog uloženog kapitala i njihove bezbednosne implikacije.

Odlaganje konačnosti (33%) - budući da je za opravdavanje prelaza između kontrolnih tačaka potrebna velika većina, napadač sa ovim udelom može sprečiti finalizaciju u konsenzusnom sloju bez direktnog uticaja na ponašanje iskrenih validatora. Odbrambeni mehanizam predstavlja aktivacija protokola curenja neaktivnosti nakon više uzastopnih epoha bez finalizacije.

Dvostruka finalnost (34%) - napadač može dvostruko atestirati sa svim svojim validatorima, pri čemu bi preostali iskreni validatori morali biti taktički podeljeni između dve grane kako bi obe bile finalizovane. Oporavak u ovom slučaju zahteva upravljanje van protokola, dok napadač biva ekonomski sankcionisan gubitkom celokupnog uloženog kapitala, što ovaj napad čini izuzetno skupim.

Kontrola budućnosti lanca (50%) - sa većinskim udelom uloga, napadač može dominirati izborom kanonske grane i uticati na uključivanje transakcija u blokove. Ublažavanje ovog napada zahteva intervenciju zajednice kroz mehanizme van protokola, uz veoma visoku ekonomsku cenu za napadača.

Kontrola prošlosti lanca (67%) - napadač može samostalno finalizovati proizvoljnu granu bez učesća iskrenih validatora, čime u potpunosti preuzima kontrolu nad konsenzusom i omogućava izvršenje dvostruke potrošnje. Jedina realna zaštita u ovom scenariju jeste ekstremno visoka ekonomska barijera za ostvarenje ovakvog napada.

Ukoliko nepošteni validatori ipak uspeju da kompromituju konsenzusni mehanizam i finalizuju verziju lanca koja je u suprotnosti sa interesima većine učesnika, Eterijum mreža se suočava sa ozbiljnim sistemskim problemom. U takvoj situaciji, mehanizmi ugrađeni u sam protokol postaju nedovoljni za oporavak, te krajnje rešenje podrazumeva oslanjanje na društveni sloj mreže kao poslednju liniju odbrane.

5.1.3.4 Društveni sloj Eterijuma

Društveni sloj (eng. *social layer*), često označavan i kao Sloj 0 (L0), obuhvata sve oblike upravljanja i donošenja odluka koji se odvijaju van formalno definisanog protokola. U ovom sloju učestvuju korisnici, validatori, programeri klijentskog softvera, istraživači i šira zajednica, čija kolektivna odluka u krajnjem slučaju određuje koju verziju lanca mreža prihvata kao legitimnu.

Napadi sa velikom težinom uloga zahtevaju izuzetno visok nivo društvene koordinacije kako bi bili neutralisani. Potencijalni odgovor zajednice može uključivati odbacivanje zlonamerne grane i prelazak na alternativnu, poštenu verziju lanca. Ovakav scenario efektivno dovodi do društveno koordinisanog razdvajanja lanca, pri čemu napadač ostaje sa finalizovanom, ali ekonomski bezvrednom verzijom sistema, budući da na njoj više ne postoji aktivna korisnička baza. Na taj način, ekonomska motivacija za sprovođenje ovakvog napada biva eliminisana.

Iako društveni sloj predstavlja snažan mehanizam krajnje odbrane, on sam po sebi nije imun na različite oblike napada. Među najznačajnije pretnje društvenom sloju Eterijuma ubrajaju se:

1. Kampanje dezinformacija sa ciljem narušavanja poverenja u Eterijum mrežu, što može dovesti do smanjenja broja učesnika i posredno do nižeg stepena decentralizacije.
2. Ciljani napadi, pritisci ili zastrašivanje usmereni ka programerima i istraživačima protokola, čime se potencijalno smanjuje broj aktivnih doprinosilaca i usporava razvoj sistema.
3. Preterano restriktivni regulatorni okviri koji mogu obeshrabriti učešće u mreži i negativno uticati na šire prihvatanje blokčejn tehnologije.
4. Infiltracija zlonamernih aktera u zajednicu sa ciljem zloupotrebe procesa donošenja odluka, produžavanja diskusija i odlaganja ključnih tehničkih unapređenja.
5. Podmićivanje ili vršenje uticaja na pojedince sa značajnom reputacijom ili formalnom ulogom u ekosistemu radi manipulacije kolektivnim odlukama.

Odbrana od napada na društveni sloj nije trivijalna i ne može se formalizovati isključivo tehničkim merama. Ipak, značajnu ulogu igra otvorena priroda Eterijum softvera (eng. *open-source*), koja omogućava transparentnost, nezavisnu verifikaciju i široko učešće zajednice. Dodatno, jasno definisana dugoročna misija i vrednosti Eterijuma doprinose jačanju otpornosti zajednice na pokušaje manipulacije.

Iz navedenih razloga, kontinuirano ulaganje u očuvanje snažnog i koherentnog društvenog sloja, zasnovanog na zajedničkim vrednostima decentralizacije, bezbednosti i otvorenosti, predstavlja ključni preduslov dugoročne stabilnosti i održivosti Eterijum protokola. Na kraju, bez obzira na tehničku sofisticiranost sistema, njegova bezbednost se u krajnjoj instanci svodi na ljude koji ga koriste, razvijaju i održavaju.

5.2 Izvršni sloj Eterijum mreže

Izvršni sloj Eterijum mreže definiše semantiku interakcije sa stanjem sistema, kao i mehanizme njegovog ažuriranja kroz izvršavanje transakcija koje mogu sadržati izvršni kod. Centralnu komponentu ovog sloja predstavlja Eterijum virtuelna mašina (EVM), koja obezbeđuje Tjuring-kompletnan model izvršavanja. Pametni ugovori, koji se najčešće pišu u programskim jezicima višeg nivoa, prevode se u EVM bajtkod (eng. *bytecode*), odnosno u niz niskonivojskih bajt-instrukcija koje se izvršavaju unutar ove virtuelne mašine.

Na ovaj način ostvaruje se determinističko, izolovano i bezbedno izvršavanje programskog koda na svim čvorovima koji učestvuju u mreži. Time se garantuje da će svi ispravni čvorovi, polazeći od istog ulaznog stanja i identičnog skupa transakcija, dospeti do istog globalnog stanja sistema.

Sa apstraktnog stanovišta, Eterijum virtuelna mašina može se posmatrati kao sistem sastavljen od sledeće tri osnovne komponente:

1. **Skladište naloga** koje predstavlja trajno stanje sistema i koje svi čvorovi čuvaju lokalno. Ono obuhvata trenutno stanje svih naloga u mreži, uključujući stanje korisnika i pametnih ugovora, i čini sastavni deo globalnog stanja Eterijum sistema.
2. **Virtuelna memorija samo za čitanje (eng. *Virtual ROM*)**, koji predstavlja prostor u kojem se smešta izvršni bajtkod pametnih ugovora u trenutku njihovog izvršavanja.
3. **Stanje mašine**, koje predstavlja izvršno okruženje u kome se izvršava bajtkod pametnih ugovora i može se posmatrati kao apstraktni virtuelni procesor koji, pored samog izvršavanja, meri i kvantifikuje računarski napor neophodan za izvršenje transakcije.

5.2.1 Skladište naloga

Promene stanja u izvršnom sloju formalizovane su pomoću Eterijumove funkcije prelaska stanja Υ (eng. *state transition function*). Ova funkcija uzima trenutno stanje sistema σ_t i validnu transakciju T , te proizvodi novo stanje σ_{t+1} , što se može formalno zapisati kao:

$$\sigma_{t+1} \equiv \Upsilon(\sigma_t, T)$$

S obzirom na to da je, prema Definiciji 4.2, blokčejn strogo hronološki uređen lanac blokova B , ažuriranje stanja Eterijum mreže odvija se na nivou blokova. U tom kontekstu, uvodi se funkcija prelaska stanja na nivou bloka Π , koja se sastoji od uzastopne primene funkcije Υ nad svim transakcijama sadržanim u bloku. Time se prethodni izraz može proširiti na sledeći način:

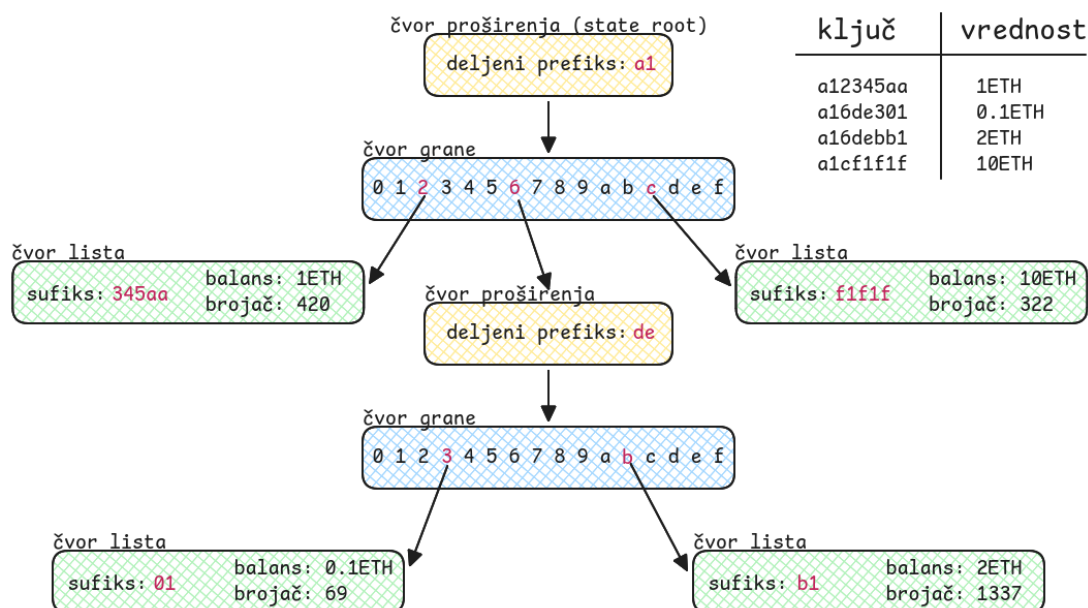
$$\sigma_{t+1} \equiv \Pi(\sigma_t, B)$$

Trenutno stanje Eterijum sistema σ (eng. *world state*) formalno se definiše kao preslikavanje između *adresa* naloga i pripadajućih *stanja* tih naloga.

$$\sigma : \text{Address}_{160} \rightarrow \text{Account}$$

Pretpostavlja se da je preslikavanje σ implementirano korišćenjem modifikovanog Merkle-Patricia stabla, koje obezbeđuje kriptografsku proverljivost integriteta podataka putem Merkle konstrukcije, kao i efikasne operacije umetanja, pretrage i brisanja sa logaritamskom složenošću u odnosu na broj

elemenata (eng. *the Practical Algorithm to Retrieve Information Coded in Alphanumeric, Patricia*) [202].



Slika 5.6: Primer uprošćenog trenutnog stanja Eterijum sistema sa 8-bitnim adresama.

Zbog toga se u svakom bloku Eterijum blokčejna čuva korenska vrednost ovog stabla (eng. *state root*), koja omogućava svim čvorovima u mreži da kriptografski verifikuju da li se nalaze u istom globalnom stanju nakon prelaska iz jednog bloka u naredni. Kako svaka promena stanja bilo kog naloga neminovno dovodi do promene korena stabla, upoređivanje isključivo ove korenske vrednosti predstavlja efikasan mehanizam za proveru konzistentnosti stanja, bez potrebe za poređenjem pojedinačnih stanja naloga tokom ažuriranja mreže.

Adrese u Eterijum mreži su fiksne dužine od 160 bita (20 bajtova) i mogu pripadati jednoj od sledećih kategorija:

1. **Nalozi u spoljnom vlasništvu** (eng. *Externally Owned Account, EOA*) predstavljaju naloge krajnjih korisnika koji poseduju sredstva i nad kojima se kontrola ostvaruje isključivo putem odgovarajućeg privatnog ključa. Gubitak privatnog ključa rezultuje trajnim gubitkom pristupa sredstvima, dok kompromitacija ključa omogućava trećim licima da se neovlašćeno predstavljaju kao vlasnici naloga. Ovi nalozi se u praksi najčešće označavaju kao adrese novčanika (eng. *wallet addresses*).
2. **Nalozi pametnih ugovora** predstavljaju naloge povezane sa pametnim ugovorima i koji se kreiraju u trenutku postavljanja pametnog ugovora na mrežu.

Stanje svakog naloga, nezavisno od toga da li se radi o nalogu u spoljnom vlasništvu ili nalogu pametnog ugovora, sastoji se od sledećih osnovnih komponenti:

1. **Balans** - količinu kriptovalute ETH koja je pridružena datom nalogu.
2. **Brojač** (eng. *nonce*) - celobrojnu vrednost koja se rastući povećava od nule i predstavlja broj transakcija iniciranih sa datog naloga. Ova vrednost služi kao mehanizam zaštite od dvostrukog trošenja, jer jednoznačno određuje redosled transakcija u istoriji izvršavanja.

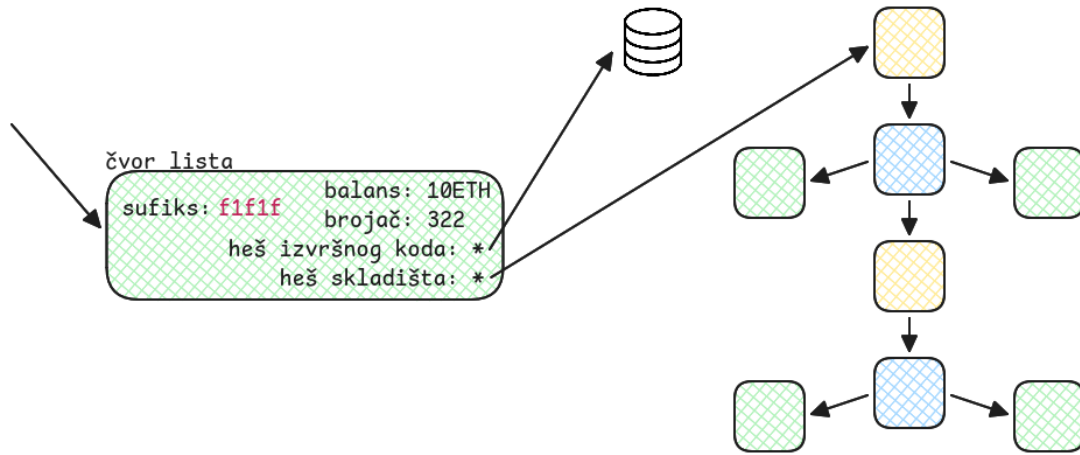
U slučaju naloga pametnog ugovora, stanje naloga može imati proizvoljnu veličinu, budući da pored osnovnih komponenti dodatno obuhvata i sledeće elemente:

3. **Heš izvršnog koda** (eng. *code hash*) - kriptografska heš vrednost nepromenljivog izvršnog koda pametnog ugovora, koja služi kao referenca odgovarajućem bajtkodu.
4. **Heš skladišta** (eng. *storage hash*) - korenski heš Merkle strukture koja predstavlja trenutno stanje trajnog skladišta pametnog ugovora.

Trenutno stanje pametnog ugovora σ_{addr} formalno se definiše kao preslikavanje između *ključeva* i *vrednosti*, pri čemu ključevi predstavljaju najmanje adresabilne skladišne jedinice (eng. *storage slots*) Eterijum virtuelne mašine. Svaki skladišni slot ima fiksnu veličinu od 32 bajta, odnosno 256 bita.

$$\sigma_{addr} : \mathbb{B}_{32} \rightarrow \mathbb{B}_{32}$$

Pretpostavlja se da je preslikavanje σ_{addr} implementirano korišćenjem modifikovanog Merkle-Patricia stabla, koje obezbeđuje kriptografsku proverljivost i efikasno ažuriranje stanja. Svaki nalog koji poseduje neprazan *heš izvršnog koda* ima pridruženo sopstveno trajno skladište, čime se implicitno identifikuje kao nalog pametnog ugovora.



Slika 5.7: Povezivanje trenutnog heša skladišta naloga pametnog ugovora sa stablom njegovog skladišta.

Shodno tome, svaka promena stanja pametnog ugovora dovodi do promene njegovog skladišta, što se reflektuje kroz izmenu odgovarajućeg heša skladišta. Ova promena se dalje propagira do korenske vrednosti Merkle-Patricia stabla globalnog stanja, čime se obezbeđuje da svaka izmena trajnog stanja bude kriptografski evidentirana na nivou celog sistema.

Eterijum virtuelna mašina, posmatrana apstraktno, pruža pametnim ugovorima mehanizam trajnog skladištenja putem instrukcije **SSTORE** [203], koji se realizuje kao skup nezavisnih skladišnih lokacija fiksne veličine od 32 bajta, kao i mehanizam čitanja skladišta pametnog ugovora instrukcijom **SLOAD** [204]. Način na koji se prosti ili složeni tipovi podataka mapiraju na ove skladišne slotove nije definisan samim EVM-om, već zavisi od kompajlera i njegovih odluka prilikom prevođenja koda iz programskog jezika višeg nivoa u odgovarajuće EVM instrukcije.

5.2.2 Virtuelna memorija samo za čitanje

Pametni ugovor napisan u programskom jeziku višeg nivoa prevodi se postupkom kompajliranja u inicijalizacioni bajtkod (eng. *init code, creation bytecode*), čija je maksimalna veličina ograničena na 48 kB [205]. Ovaj bajtkod obuhvata konstruktorski bajtkod (eng. *constructor code*) i bajtkod za izvršavanje (eng. *runtime bytecode*).

Prilikom postavljanja ugovora na mrežu (eng. *deployment*), transakcija sadrži instrukciju **CREATE** ili **CREATE2** [206], čijim se izvršavanjem inicijalizuje stanje ugovora kroz izvršavanje konstruktorskog bajtkoda. Po završetku ove faze, konstruktorski bajtkod se odbacuje, dok se izvršni bajtkod trajno smešta u nepromenljivi i logički izdvojeni segment sistema, a njegov heš se upisuje u stanje naloga pametnog ugovora kao referenca na taj kod.

Dobijen izvršni bajtkod ostaje trajno vezan za nalog pametnog ugovora i ne može se menjati tokom njegovog životnog veka, pri čemu je pristup ovom bajtkodu omogućen isključivo putem heša izvršnog koda u stanju pametnog ugovora. Prilikom izvršavanja transakcije pametnog ugovora, ovaj bajtkod se učitava u virtuelnu memoriju samo za čitanje (eng. *Virtual ROM*), kojoj se pristupa tokom daljeg izvršavanja instrukcija transakcije.

Prilikom analize postavljanja pametnih ugovora na mrežu, na osnovu heša izvršnog koda uočeno je da približno četvrtina ugovora koristi identičan bajtkod kao već postojeći ugovori [207]. Ova pojava ukazuje na potrebu za deduplikacijom koda (eng. *code deduplication*), budući da čvorovi u praksi ne dupliraju bajtkod u skladištu, dok korisnici i dalje snose pun trošak njegovog postavljanja kao da se radi o jedinstvenom kodu. U cilju rešavanja ovog problema, predloženi su mehanizmi kojima se prilikom postavljanja ugovora eksplicitno referencira već postojeći bajtkod u mreži, čime se izbegavaju suvišni troškovi skladištenja, što je trenutno predmet otvorenih diskusija o unapređenju protokola [208].

Korisnici prilikom interakcije sa pametnim ugovorom u polje podataka transakcije enkodiraju prva četiri bajta heša potpisa funkcije (eng. *function selector*), kao i pripadajuće argumente u odgovarajućoj bajtovskoj reprezentaciji, što čini podatke za poziv (eng. *calldata*) transakcije.

Ukoliko tražena funkcija ne postoji, a definisana je podrazumevana funkcija (eng. *fallback*), ona se poziva umesto nje i najčešće se koristi za obradu neočekivanih poziva i očuvanje konteksta pri delegiranju izvršavanja između pametnih ugovora [209]. U suprotnom, ukoliko ni takva funkcija nije definisana, izvršavanje transakcije rezultuje neuspehom.

Pre početka izvršavanja, kontrola toka prolazi kroz dispečersku logiku (eng. *dispatcher*), ugrađenu u bajtkod ugovora, koja na osnovu prosleđenih podataka za poziv izdvaja selektor funkcije i, putem uslovnih skokova, usmerava izvršavanje na odgovarajući deo bajtkoda.

Međutim, direktno pozivanje funkcija u obliku bajtkoda nije ni ljudski čitljivo niti praktično za upotrebu, zbog čega značajnu ulogu ima aplikacioni binarni interfejs (eng. *Application Binary Interface, ABI*). ABI definiše imena funkcija, njihove potpise i tipove argumenata u strogo strukturisanom formatu, čime uspostavlja formalni dogovor između pametnog ugovora i spoljašnjih korisnika ili aplikacija.

U suštini, ABI deluje kao prevodilački sloj između ljudski čitljivog interfejsa pametnog ugovora i njegove niskonivojske bajtovske reprezentacije koja se izvršava na blokčejnu, omogućavajući pouzdanu i transparentnu interakciju bez potrebe za ručnim enkodiranjem ili dekodiranjem transakcionih podataka.

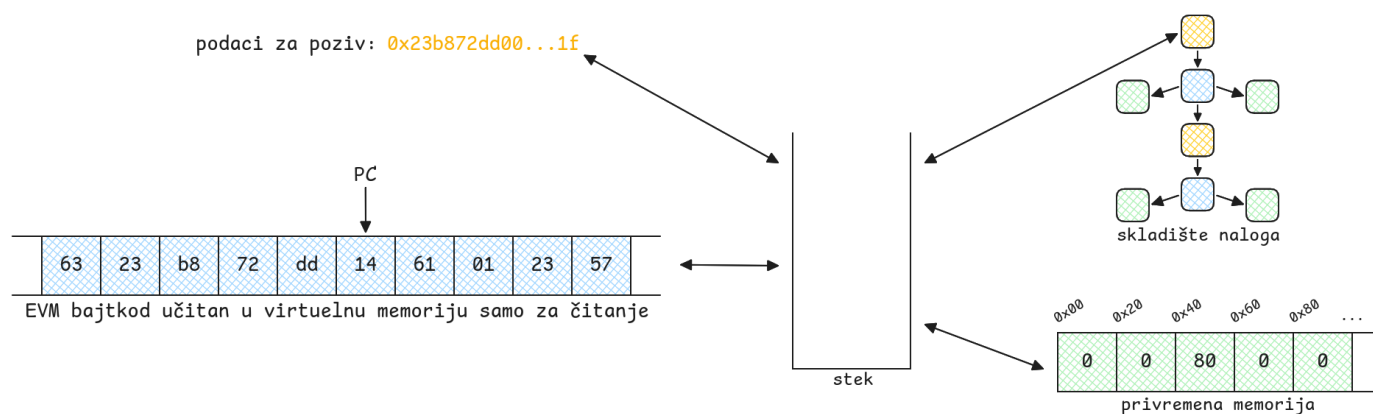
5.2.3 Stanje mašine

Postavljanjem pokazivača u virtualnoj memoriji samo za čitanje na adresu bajt-instrukcije od koje započinje izvršavanje funkcije pametnog ugovora implicitno se određuje odgovarajući segment koda (eng. *code segment*) iz kojeg se dalje odvija izvršavanje date transakcije. U tom trenutku inicijalizuje se stanje mašine Eterijum virtuelne mašine, koje se uništava po završetku izvršavanja transakcije, zbog čega ima isključivo privremeni karakter.

Stanje Eterijum virtuelne mašine obuhvata sledeće komponente:

1. **Brojač programa** (eng. *Program Counter, PC*) - označava poziciju (adresu) sledećeg bajta u bajtkodu koji će biti interpretiran kao kod instrukcije ili njegov operand.
2. **Stek** - struktura podataka koja omogućava pristup isključivo elementu na vrhu (eng. *last in first out, LIFO*), maksimalne dubine od 1024 elementa, pri čemu je svaki element veličine jedne skladišne reči.
3. **Privremena memorija** - linearni niz bajtova sa direktnim pristupom elementima, koji služi kao pomoćni prostor za izvođenje proračuna tokom izvršavanja transakcije.
4. **Preostali gas** - kvantitativna mera raspoloživih računarskih resursa izraženih u jedinicama gasa, koja određuje da li transakcija poseduje dovoljno sredstava za svoje kompletno izvršavanje.

S obzirom na to da je bajtkod pametnog ugovora predstavljen kao niz bajtova, u kojem pojedini bajtovi predstavljaju operacione kodove, dok drugi mogu predstavljati njihove operande, izvršavanje koda može se modelovati kao sekvencijalni proces vođen brojačem programa. U tom kontekstu, EVM se može posmatrati kao stek-orientisana virtuelna mašina (eng. *stack-based virtual machine*) [210], u kojoj instrukcije implicitno koriste operande sa vrha steka i rezultate vraćaju nazad na stek. Ovakav model omogućava izvršavanje instrukcija koje manipulišu različitim segmentima stanja, uključujući bajtkod ugovora, podatke za poziv, privremenu memoriju i trajno skladište.



Slika 5.8: Kontekst izvršavanja transakcije u EVM okruženju.

Ukoliko se vrednost brojača programa odnosi na kod instrukcije, proverava se da li je raspoloživa dovoljna količina gasa za njegovo izvršavanje, nakon čega se brojač ažurira. Proces se nastavlja sve dok se ne dostigne kraj bajtkoda, ne izvrši instrukcija `STOP` ili `RETURN`, ili dok ne nastupi greška tokom izvršavanja.

U slučaju da transakcija sadrži više internih poziva, odnosno interakciju sa više pametnih ugovora, za svaki takav poziv inicijalizuje se zasebno stanje mašine. Međutim, radi očuvanja konteksta izvršavanja unutar iste transakcije, uvedeno je i **privremeno skladište** [211] (eng. *transient storage*), čiji je životni vek ograničen na trajanje transakcije.

Gas predstavlja obračunsku jedinicu kojom se meri količina računarskih resursa potrebnih za izvršavanje transakcije u Eterijum mreži. Ukupna transakciona naknada, u smislu Definicije 4.4, određuje se na osnovu ukupne potrošnje gasa tokom izvršavanja transakcije i cene gasa koju korisnik navodi prilikom njenog slanja. Na taj način se obezbeđuje da upotreba računarskih resursa mreže ima eksplicitnu ekonomsku cenu.

Cena gasa nije konstantna i može se menjati tokom vremena u zavisnosti od opterećenja mreže. Prilikom slanja transakcije, korisnik definiše cenu gasa koju je spreman da plati za korišćenje računarskih resursa mreže. Ova cena se izražava u denominaciji osnovne valute sistema, pri čemu važi odnos $1 \text{ gwei} = 10^{-9} \text{ ETH}$.

U Eterijum sistemu standardni tip transakcije koristi mehanizam osnovne naknade [212] (eng. *base fee*), koja predstavlja minimalnu cenu gasa po jedinici resursa koju korisnik mora da plati kako bi transakcija bila uključena u blok. Ovaj mehanizam zasniva se na konceptima ponude i potražnje:

1. **Ponuda** predstavlja ukupnu količinu računarskih resursa koju Eterijum mreža može da potroši u jednom koraku napredovanja sistema, odnosno maksimalnu količinu gasa po bloku. Ova vrednost je protokolski ograničena i dinamički prilagođavana, te u trenutnoj implementaciji iznosi 60 miliona gasa [213], što omogućava ostvarivanje performansi reda veličine oko 40 TPS.
2. **Potražnja** odgovara ukupnoj količini gasa utrošenoj na izvršavanje svih transakcija u prethodnom bloku, koja se poredi sa ciljnom vrednošću od 50% iskorišćenosti raspoložive ponude gasa.

Ukoliko je iskorišćenost gasa u prethodnom bloku veća od 50%, osnovna naknada se povećava, dok se u suprotnom slučaju smanjuje. Promena osnovne naknade ograničena je na maksimalno 12.5% po bloku, čime se obezbeđuje stabilnost sistema. Ovakav eksponencijalni mehanizam prilagođavanja (Tabela 5.1) onemogućava dugotrajno zadržavanje visoke popunjenosti blokova i doprinosi ekonomskoj održivosti mreže.

Tabela 5.1: Eksponencijalni rast cene osnovne naknade prilikom velike opterećenosti mreže

Broj bloka	Iskorišćenost mreže	Povećanje naknade	Trenutna osnovna naknada
1	50%	0%	100 gwei
2	100%	12,5%	100 gwei
3	100%	12,5%	112.5 gwei
4	100%	12,5%	126.6 gwei
5	100%	12,5%	142.4 gwei
...	...	12,5%	...
30	100%	12,5%	2705.6 gwei
...	...	12,5%	...
100	100%	12,5%	10302608.6 gwei

Izvor: <https://ethereum.org/developers/docs/gas/>

Izračunavanje ukupne naknade za gas transakcije se vrši pomoću sledeće formule:

$$Ukupna\ transakciona\ naknada = Količina\ gasa \cdot (Osnovna\ naknada + Prioritetna\ naknada)$$

gde je prioritetna naknada opcioni bakšiš za transakcije višeg prioriteta.

5.3 Transakcije i njihova obrada u Eterijumu

U Eterijum sistemu transakcije mogu inicirati isključivo nalozi kojima upravljaju korisnici. Pametni ugovori ne mogu samostalno inicirati transakcije, već se njihovo izvršavanje pokreće isključivo kao posledica transakcije koju inicira korisnik. Tokom izvršavanja pametnog ugovora moguće je generisati interne pozive ka drugim ugovorima ili nalogima, ali se oni ne smatraju nezavisnim transakcijama već delom izvršavanja iste transakcije.

Prilikom slanja transakcije, korisnik je dužan da kriptografski potpiše skup sledećih polja:

1. **od** - adresa pošiljaoca koja potpisuje transakciju,
2. **do** - adresa primaoca i ukoliko se radi o nalogu u spoljnom vlasništvu vrši se prenos vrednosti, dok se u slučaju pametnog ugovora izvršava njegov kod,
3. **potpis** - kriptografski dokaz kojim pošiljalac potvrđuje da je transakciju autorizovao svojim privatnim ključem,
4. **brojač** - redni broj transakcije inicirane sa datog naloga, koji sprečava ponavljanje transakcija,
5. **vrednost** - iznos u ETH koji se prenosi sa pošiljaoca na primaoca,
6. **ulazni podaci** - opciono polje koje se koristi za prosleđivanje dodatnih podataka, najčešće prilikom interakcije sa pametnim ugovorima,
7. **ograničenje gasa** (eng. *gasLimit*) - maksimalna količina gasa koju transakcija može potrošiti tokom izvršavanja,
8. **cena gasa** - cena koju je korisnik spreman da plati po jedinici gasa.

Nakon uspešnog izvršavanja transakcije generiše se priznanica transakcije (eng. *transaction receipt*), koja sadrži informacije o ishodu izvršavanja. Priznanice svih transakcija u okviru jednog bloka organizuju se u Merkle-Patricia strukturu, pri čemu se u zaglavlju bloka čuva isključivo njen korenski heš (eng. *receiptsRoot*). Same priznanice se ne skladište eksplicitno u strukturi bloka, već ih čvorovi čuvaju u lokalnim bazama podataka, uz mogućnost pristupa na osnovu identifikatora transakcije.

Tokom izvršavanja transakcije u kontekstu pametnog ugovora, ugovor može emitovati događaje (eng. *events*) koji opisuju relevantne promene nastale tokom izvršavanja. Ovi događaji nisu dostupni za korišćenje unutar izvršnog modela blokčejn sistema, već su namenjeni spoljnim aplikacijama za praćenje i/ili pretplaćivanje. Svaki događaj može sadržati i skup indeksa (eng. *topics*), koji omogućavaju efikasno filtriranje i pretraživanje događaja u okviru istorije izvršavanja pametnog ugovora.

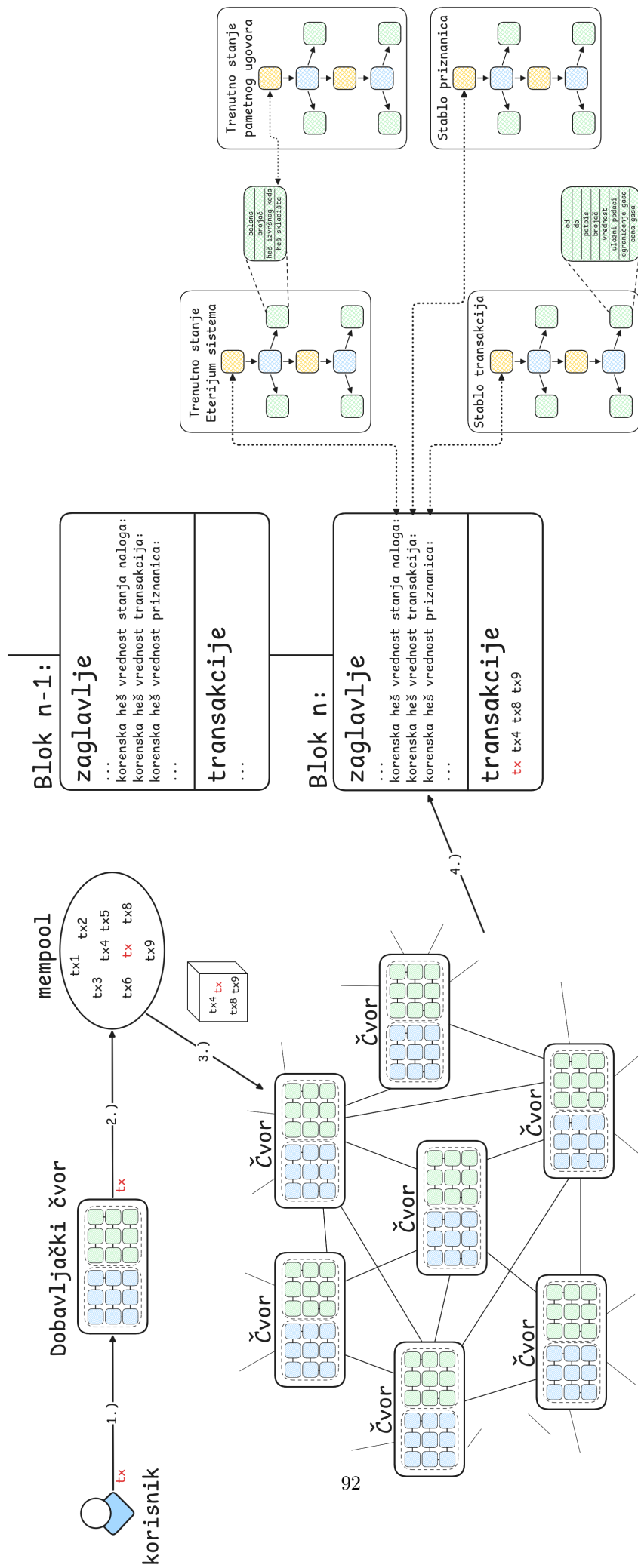
Eterijum je inicijalno podržavao samo jedan format transakcija, ali je sistem vremenom evoluirao kako bi omogućio uvođenje novih funkcionalnosti bez narušavanja kompatibilnosti sa postojećim formatima. Ovo je postignuto uvođenjem tipiziranih transakcija, pri čemu se svakom tipu dodeljuje numerička oznaka u opsegu od 0 do 127, koja prethodi ostatku transakcionih podataka [214].

U zavisnosti od vrednosti tipa, transakcije se klasifikuju na sledeći način:

1. **Tip 0** - originalni format transakcije bez eksplicitnog prefiksa tipa, koji se implicitno tumači kao tip 0,
2. **Tip 1** - transakcije koje uvode dodatno polje sa listama ugovora i skladišnih jedinica kojima će se pristupati tokom izvršavanja [215], čime se omogućava smanjenje troškova gasa,
3. **Tip 2** - transakcije koje uvode mehanizam osnovne naknade i dodatna polja za maksimalnu prioritetnu naknadu (eng. *maxPriorityFeePerGas*) i maksimalnu ukupnu naknadu (eng. *maxFeePerGas*), čime se postiže veća efikasnost i fleksibilnost,
4. **Tip 3** - transakcije koje omogućavaju privremeno skladištenje velikih binarnih objekata [216] (eng. *Binary Large Objects, BLOB*), čime se značajno unapređuje dostupnost podataka i skalabilnost mreže,
5. **Tip 4** - transakcije koje omogućavaju da se nalog u spoljnom vlasništvu privremeno ponaša kao pametan ugovor [217], bez narušavanja njegove osnovne funkcionalnosti, što predstavlja važan korak ka dugoročnim ciljevima razvoja apstrakcije naloga u Eterijum sistemu.

Imajući u vidu prethodno opisane komponente sistema, sada se može sagledati kompletan tok jedne transakcije u Eterijum mreži:

1. Korisnik bira odgovarajući tip transakcije, popunjava neophodna polja, kriptografski je potpisuje i prosleđuje dobavljačkom čvoru (eng. *provider*) Eterijum mreže.
2. Nakon inicijalne validacije formata i osnovnih pravila ispravnosti, transakcija se smešta u lokalni skup nepotvrđenih transakcija (eng. *mempool*). Zatim se putem ogovaračkog protokola propagira ka ostalim čvorovima mreže, koji je po prijemu smeštaju u sopstvene lokalne skupove nepotvrđenih transakcija.
3. U narednom slotu, validator izabran za predlagača bloka odlučuje da li će korisnikova transakcija biti uključena u predloženi blok, pri čemu na ovu odluku značajno utiče visina cene gasa koju je korisnik spreman da plati po jedinici resursa.
4. Predloženi blok se propagira kroz mrežu, pri čemu ostali validatori proveravaju njegovu ispravnost, izvršavaju transakcije, ažuriraju lokalno stanje sistema i nastavljaju proces glasanja i finalizacije u skladu sa pravilima konsenzusa.



Slika 5.9: Kompletan tok obrade transakcije u Eterijum mreži.

Glava 6

Metodologija optimizacije pametnih ugovora za e-trgovinu

Neoptimizovane funkcije pametnih ugovora mogu značajno povećati količinu gasa potrebnu za njihovo izvršavanje usled uvećanog računarskog napora, naročito u uslovima povećanog opterećenja mreže [218]. Ovakvo ponašanje negativno utiče na korisničko iskustvo, ali i na ukupne performanse blokčejn sistema, s obzirom na to da je veličina blokova ograničena ukupnom količinom gasa koja se može potrošiti u jednom napredovanju sistema. Stoga se kao primarni cilj nameće minimizacija potrošnje gasa tokom izvršavanja funkcija pametnih ugovora.

Posebno značajan faktor u tom kontekstu predstavlja trajno skladištenje podataka na blokčejnu. Svaki podatak upisan u skladište mora biti lokalno sačuvan od strane svih čvorova mreže, čime se direktno povećava ukupna količina podataka koju sistem mora da održava. Na primer, upis jedne skladišne jedinice veličine 32 bajta u sistemu sa milion čvorova rezultuje ukupnim replikovanim prostorom od približno

$$32 \text{ bytes} \cdot 1,000,000 \approx 30 \text{ MB}$$

što ilustruje značajne prostorne i infrastrukturne troškove čak i za minimalne upise.

Kako su operacije čitanja i pisanja u skladište među najskupljim operacijama u pogledu potrošnje gasa, fokus optimizacije usmeren je na njihovu minimizaciju. Sve ostale operacije koje se izvršavaju tokom izvršavanja funkcije imaju znatno manji uticaj na ukupnu potrošnju gasa i u daljoj analizi biće aproksimativno modelovane radi pojednostavljenja formalnog zapisa. Iz navedenih razloga, metodologija optimizacije pametnih ugovora u ovom radu primarno je usmerena na smanjenje upotrebe trajnog skladišta u cilju smanjenja troškova koje snose korisnici.

6.1 Troškovi pristupa skladištu i njihov uticaj na cenu izvršavanja

U Eterijum sistemu svaka transakcija koju korisnik podnosi poseduje osnovni, tzv. unutrašnji gas (eng. *intrinsic gas*), koji predstavlja minimalni trošak obrade transakcije nezavisno od njenog izvršnog sadržaja. S obzirom na to da dužina ulaznih podataka može varirati, potrošnja gasa direktno je povezana sa količinom podataka i računarskim naporom potrebnim za njihovu obradu.

Osnovna cena transakcije iznosi 21.000 jedinica gasa i obuhvata računarske resurse potrebne za

uključivanje transakcije u blok, uključujući verifikaciju potpisa, validaciju transakcije, ažuriranje stanja, upis u blok i propagaciju kroz mrežu. Dodatni trošak zavisi od veličine ulaznih podataka prosleđenih u okviru transakcije, pri čemu se za svaki bajt ulaznih podataka plaćaju 4 jedinice gasa ukoliko je bajt jednak nuli, odnosno 16 jedinica gasa za svaki nenulti bajt [219].

Tokom izvršavanja transakcije, Eterijum virtuelna mašina izvršava instrukcije pametnog ugovora, pri čemu operacije koje čitaju ili menjaju stanje ugovora značajno doprinose ukupnoj ceni izvršavanja koju snosi korisnik. Radi optimizacije, sistem uvodi mehanizam keširanja skladišnih lokacija u okviru jedne transakcije [203], čime se umanjuje trošak višestrukih pristupa istom skladišnom slotu. Ovaj mehanizam uveden je kao odgovor na ranije napade uskraćivanja usluge, imajući u vidu da su operacije pristupa stanju dominantno određene troškom ulazno-izlaznih operacija [220].

Prvi pristup skladišnoj jedinici u toku transakcije označava se kao hladan pristup (eng. *cold access*), nakon čega se ta skladišna jedinica smatra toplom (eng. *warm access*) za preostalo trajanje iste transakcije. Cena hladnog pristupa iznosi 2.100 jedinica gasa, dok svaki naredni topli pristup istoj skladišnoj jedinici košta 100 jedinica gasa.

Operacija upisa u skladište, realizovana instrukcijom **SSTORE**, ima različite cene u zavisnosti od prethodnog i novog stanja skladišne jedinice:

1. Ukoliko je skladišna jedinica inicijalno prazna (sadrži nultu vrednost), a instrukcija **SSTORE** je postavlja na nenultu vrednost, povećava se ukupna količina trajnih podataka u sistemu, te ova operacija košta 20.000 jedinica gasa,
2. Ukoliko skladišna jedinica već sadrži nenultu vrednost i u okviru transakcije se menja u drugu nenultu vrednost, trošak iznosi 2.900 jedinica gasa,
3. Svaka naredna promena iste skladišne jedinice u okviru iste transakcije dodatno košta 100 jedinica gasa,
4. Ukoliko se skladišna jedinica sa nenultom vrednošću postavi na nultu vrednost, korisniku se odobrava povraćaj od 4.800 jedinica gasa, čime se umanjuje cena trenutne transakcije.

U daljoj analizi, prve dve navedene cene označavaćemo kao **SSTORE(1)** i **SSTORE(2)**, dok će se izraz „ako nije nula” odnositi na slučaj kada skladišna jedinica pre početka transakcije sadrži nenultu vrednost, pa se primenjuje cena **SSTORE(2)**, jer bi se u suprotnom primenjivala cena **SSTORE(1)**. Pri-kazane cene podrazumevaju topli pristup skladištu, dok u slučaju hladnog pristupa ukupni troškovi iznose 22.100, odnosno 5.000 jedinica gasa.

Zbog postojanja mehanizma povraćaja gasa prilikom oslobađanja skladišnih lokacija, programeri su podstaknuti da razvijaju pametne ugovore koji primenjuju praksu efikasnog upravljanja stanjem [221], odnosno da uklanjaju podatke koji više nisu potrebni. Ovakav pristup doprinosi smanjenju ukupnih troškova izvršavanja, čime se posredno unapređuje korisničko iskustvo.

S obzirom na to da troškove izvršavanja transakcija snose korisnici koji ih iniciraju, trošak postavljanja pametnog ugovora na mrežu snosi nalog koji vrši njegovo inicijalno kreiranje. Tokom transakcije za postavljanje ugovora obračunavaju se sledeće komponente:

1. Osnovna cena unutrašnjeg gasa transakcije,
2. Trošak instrukcije **CREATE** ili **CREATE2**, sa osnovnom cenom od 32.000 jedinica gasa,
3. Dodatni trošak inicijalizacionog bajtkoda od 2 jedinice gasa po svakoj grupi od 32 bajta [205],

4. Trošak izvršavanja konstruktorskog bajtkoda, pri čemu se operacije upisa u skladište naplaćuju prema ceni `SSTORE(1)` sa toplim pristupom,
5. Trošak trajnog smeštanja izvršnog bajtkoda na mreži, po ceni od 200 jedinica gasa po bajtu.

Usled ranijih zloupotreba mehanizma povraćaja gasa putem tzv. tokena za gas [222], Eterijum protokol je onemogućio povraćaj gasa prilikom uništavanja izvršnog bajtkoda pametnog ugovora instrukcijom `SELFDESTRUCT`.

6.2 Modelovanje vlasništva nad digitalnim entitetima na blokčejnu u sistemima e-trgovine

Nezamenljivi tokeni (eng. *non-fungible token, NFT*) predstavljaju digitalne tokene sa jedinstvenim identifikatorima koji se trajno evidentiraju na blokčejnu, čime se unapređuju bezbednost i proverljivost digitalne imovine. Prvi konceptualni oblici nezamenljivih tokena pojavili su se na Bitcoin mreži kroz ideju tzv. obojenih novčića [223] (eng. *colored coins*), a zatim i kroz platformu *Counterparty* [224]. Iako su ovi rani sistemi bili eksperimentalne prirode, sa ograničenim alatima i nedovoljno razvijenim ekosistemom, demonstrirali su mogućnost predstavljanja jedinstvenih digitalnih entiteta na blokčejnu.

Eterijum je standardizovao koncept nezamenljivih tokena kroz ERC-721 standard [225], kojim se definišu struktura i osnovna funkcionalnost NFT-ova u okviru jednog pametnog ugovora. Ovaj standard uspostavlja šemu za kreiranje jedinstvenih i nedeljivih tokena, čime je omogućena njihova masovna primena i interoperabilnost unutar ekosistema. Do maja 2024. godine ukupna tržišna vrednost NFT projekata dostigla je približno 69 milijardi američkih dolara [226], što potvrđuje njihov značaj i široku prihvaćenost.

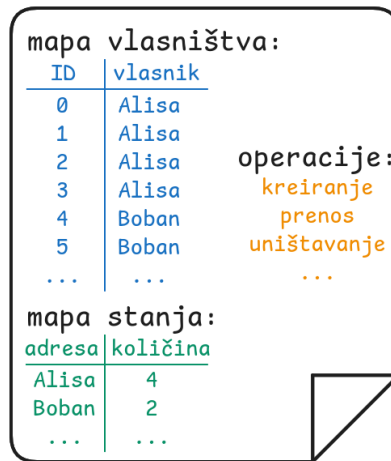
Osnovni NFT standard optimizovan je u skladu sa troškovnim modelom Eterijum mreže i kontinuirano se prilagođava kroz unapređenja protokola. Vremenom su razvijeni i alternativni standardi koji zadržavaju koncept jedinstvenog identifikatora, ali uvode implicitne modele reprezentacije vlasništva radi smanjenja troškova skladištenja i izvršavanja transakcija, kao i standardi koji kombinuju zamenljive i nezamenljive tokene u okviru jedinstvenog modela.

U nastavku rada razmatraju se isključivo standardi za nezamenljive tokene koji počivaju na eksplicitnoj reprezentaciji vlasništva, gde se svaki token jednoznačno povezuje sa adresom vlasnika putem jednostrukog mapiranja:

$$\text{mapping}(\text{uint256 tokenId} \Rightarrow \text{address}) \text{ private } _owners;$$

Ovakva struktura pogodna je za primenu u sistemima e-trgovine, gde svaki proizvod ili usluga mora posedovati jedinstveni identifikator. Imajući u vidu da EVM operiše nad skladišnim jedinicama veličine 256 bita, dok adrese zauzimaju 160 bita, preostalih 96 bita moguće je iskoristiti za kompaktno enkodiranje dodatnih informacija o proizvodu ili usluzi unutar iste skladišne jedinice.

U ovom radu primenjuje se detaljna analiza osnovnih operacija, kreiranja (eng. *minting*), prenosa i uništavanja (eng. *burn*), kada su dodatne informacije enkodirane u preostalih 96 bita, pri čemu se upoređuje potrošnja gasa u različitim scenarijima ulaza. Analiza obuhvata procenu prednosti i ograničenja predloženog pristupa, uz očuvanje funkcionalne kompatibilnosti i bezbednosnih svojstava originalnih standarda.



Slika 6.1: Struktura pametnog ugovora za nezamenljive tokene u kojem se vodi evidencija o vlasništvu nad identifikatorima, stanjima adresa i osnovnim operacijama koje će biti analizirane.

U prethodnom radu [227] fokus je bio na modelovanju elektronskih kupona kao specifične instance digitalnih entiteta u sistemima elektronske trgovine, imajući u vidu da savremeno tržište u velikoj meri zavisi od njihove primene. U tom radu demonstrirano je da se značajne uštede u potrošnji gasa ostvaruju enkodiranjem svih relevantnih informacija o kuponu unutar jedne skladišne jedinice. Time je unapređen pristup ranijim rešenjima koja su za svaki novi kupon instancirala zaseban pametni ugovor, što je izrazito skladišno i infrastrukturno neefikasno.

Iako su eksperimentalna merenja sprovedena nad modelom elektronskih kupona, predložena metodologija nije ograničena isključivo na tu primenu. U zaključku tog rada eksplicitno je istaknuto da su sprovedena merenja relevantna za sve platforme elektronske trgovine koje svoje proizvode ili usluge predstavljaju putem NFT-ova čiji se atributi mogu kompaktно enkodirati u okviru jedne skladišne jedinice. U tom smislu, **kuponi predstavljaju reprezentativan studijski slučaj**, dok su izvedeni zaključci primenljivi na širi spektar NFT-zasnovanih modela u elektronskoj trgovini.

Predložena metodologija primenljiva je na svim blokčejn sistemima zasnovanim na Eterijum virtualnoj mašini, imajući u vidu njenu široku zastupljenost i standardizovan model skladišta. Dodatno, koncept je prilagodljiv i drugim sinhronim blokčejn sistemima koji koriste sličan model izvršavanja i upravljanja stanjem.

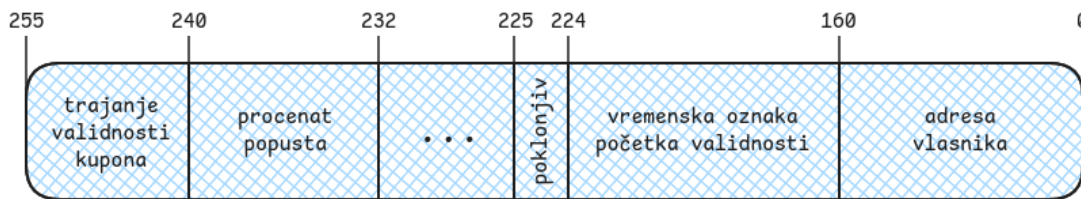
6.3 Enkodirana reprezentacija entiteta sa jedinstvenim identifikatorom u okviru jedne skladišne jedinice

Kako bi se elektronski kuponi prilagodili NFT funkcionalnostima, sve relevantne informacije o kuponu enkodirane su tako da zauzimaju tačno jednu skladišnu jedinicu. Ovo se postiže primenom bitskih operacija ili pažljivim pakovanjem podataka u složenu strukturu [228]. Kada susedna polja unutar strukture zahtevaju ukupno 256 bita ili manje, kompajler vrši optimizaciju i kombinuje više vrednosti u jednu skladišnu reč od 256 bita.

Na ovaj način obezbeđuje se da se prilikom upisa podataka izvršava samo jedna **SSTORE** operacija, dok je za čitanje svih atributa kupona dovoljna jedna **SLOAD** operacija, čime se minimizuje broj pristupa trajnom skladištu.

U okviru ovakvog modela, kupon sadrži sledeće informacije:

1. Adresu vlasnika kupona (160 bita),
2. Početnu vremensku oznaku u *Unix* formatu koja definiše trenutak od kog kupon postaje validan (64 bita),
3. Skup od osam binarnih promenljivih, svaka veličine jednog bita. U demonstracionom modelu jedna promenljiva označava da li je kupon „poklonjiv”, što omogućava njegov prenos između korisnika. Druge promenljive mogu imati funkcionalnosti poput „tržišno prenosiv”, čime bi se omogućila sekundarna trgovina (ukupno 8 bita),
4. Celobrojnu vrednost koja predstavlja procenat popusta (8 bita),
5. Parametar koji definiše trajanje validnosti kupona nakon početne vremenske oznake. Ova vrednost izražena je u jednakim vremenskim intervalima, koji se definišu prilikom postavljanja ugovora (16 bita).



Slika 6.2: Reprezentacija informacija elektronskog kupona enkodiranih u jednu EVM skladišnu jedinicu.

Ovakvo gusto enkodiranje podataka nužno uvodi određena ograničenja. Na primer, korišćenje 8 bita za procenat popusta onemogućava reprezentaciju decimalnih vrednosti. Slično tome, izbor veličine vremenskog intervala direktno utiče na maksimalno moguće trajanje važenja kupona i ukoliko je jednak jednom danu, maksimalna dužina važenja iznosi približno ≈ 20 godina. Iako je moguće primenom bitskih operatora precizno kontrolisati raspodelu proizvoljnog broja bitova, uključujući vrednosti koje nisu deljive sa 8, u ovom radu je takav pristup izostavljen radi jednostavnosti implementacije i preglednosti formalne analize.

Algoritam 6.1 Implementacija konstruktora elektronskih kupona u skladu sa NFT standardom

```

1: constructor(name, symbol, blockDuration)
2:   _name = name
3:   _symbol = symbol
4:   _BLOCK_DURATION = blockDuration
5:   _currentIndex = _startTokenId()
6: end constructor

```

Prilikom postavljanja pametnog ugovora na mrežu, NFT standardi čuvaju metapodatke, ime i simbol, radi jednostavnog razlikovanja kolekcija, jer bi se bez njih kolekcija prikazivala kao adresa ugovora, što nije ljudski čitljivo. Dodatno, radi prilagođavanja standarda implementaciji elektronskih kupona, uvedena je dodatna promenljiva koja definiše trajanje osnovnog vremenskog intervala, označena kao `_BLOCK_DURATION`, koja se takođe inicijalizuje prilikom postavljanja pametnog ugovora na mrežu i koristi za određivanje ukupnog trajanja validnosti kupona. Inicijalizacija ovih

promenljivih, kao i početnog identifikatora prvog tokena (čija je podrazumevana vrednost 0), vrši se u konstruktoru pametnog ugovora, kao što je prikazano u Algoritmu 6.1.

Svi razmatrani NFT standardi dele istu osnovnu logiku za ključne operacije. Algoritam 6.2 prikazuje pseudokod operacija kreiranja, prenosa i uništavanja, pri čemu su promenljive kupona grupisane u jednu skladišnu jedinicu, kao što je prethodno opisano. Pseudokod je zasnovan na programskom jeziku *Solidity*, kao najzastupljenijem jeziku za razvoj aplikacija nad EVM-om. Na početku svake funkcije definišu se uslovi ispravnosti putem naredbe *require*, u skladu sa principima bezbednog dizajna pametnih ugovora i ustaljenom praksom u industriji.

Algoritam 6.2 Osnovna logika funkcija kreiranja, prenosa i uništavanja koje e-kupon implementira primenom odgovarajućih NFT standarda

```

1: function MINT(to, quantity, startTimestamp, gitable, percentage, numOfBlocks)
2:   require(to  $\neq$  address(0))
3:   require(quantity > 0)
4:   require(percentage > 0 and percentage  $\leq$  100)
5:   require(numOfBlocks > 0 and numOfBlocks <  $2^{16}$ )
6:   _owners[currentIndex] = Coupon(to, startTimestamp, gitable, percentage, numOfBlocks)
7:   _balances[to] += quantity
8:   currentIndex += quantity
9: end function

1: function TRANSFER(from, to, id)
2:   owner = ownerOf(id)
3:   require(owner == from)
4:   require(to  $\neq$  address(0))
5:   require(_owners[id].gitable == true)
6:   _balances[from]--
7:   _balances[to]++
8:   _owners[id].owner = to
9: end function

1: function BURN(id)
2:   owner = ownerOf(id)
3:   coupon = owners[id]
4:   require(coupon.startTimestamp + coupon.numOfBlock * _BLOCK_DURATION  $\geq$ 
    block.timestamp)
5:   _balances[owner]--
6:   delete owners[id]
7: end function

```

U narednim pododeljcima dat je hronološki pregled razvoja NFT standarda. Sistematski su opisani nedostaci identifikovani u svakom standardu, koji su poslužili kao podsticaji za razvoj narednih. Za svaki standard predstavljena je logika operacija kreiranja, prenosa i uništavanja, sa detaljnom analizom instrukcija koje pristupaju skladištu, jer one predstavljaju glavne faktore koji utiču na potrošnju gasa. Uticaj ostalih operacija je minimalan i biće aproksimiran po potrebi.

6.3.1 ERC-721

ERC-721 predstavlja osnovni standard za implementaciju nezamenljivih tokena, definišući interfejs i minimalni skup funkcionalnosti neophodnih za njihovo kreiranje, praćenje vlasništva i prenos. Osnovna logika operacija kreiranja, prenosa i uništavanja ostaje konceptualno identična opštem modelu prikazanom u Algoritmu 6.2, te se u ovom pododeljku fokus stavlja isključivo na specifičnosti

implementacije i njihove implikacije na pristup skladištu i potrošnju gasa.

Standard uspostavlja model u kojem se vlasništvo nad tokenima evidentira putem mape vlasnika `_owners`, koja preslikava jedinstveni identifikator tokena na adresu njegovog vlasnika, dok se ukupan broj tokena u vlasništvu određene adrese prati putem mape stanja `_balances`. Na osnovu ovih struktura moguće je efikasno odrediti vlasnika proizvoljnog tokena, kao i ukupan broj tokena koje određeni korisnik poseduje.

Originalni ERC-721 standard omogućava kreiranje tokena sa proizvoljno zadatim identifikatorima, pod uslovom da dati identifikator nije prethodno dodeljen. Međutim, radi usklađivanja eksperimentalnih uslova sa naprednijim standardima koji se razmatraju u nastavku rada, ovaj mehanizam je modifikovan uvođenjem promenljive `_currentIndex`, koja označava identifikator narednog tokena koji će biti kreiran. Na ovaj način obezbeđuje se sekvencijalna dodela identifikatora i omogućava konzistentno poređenje potrošnje gasa između različitih standarda.

6.3.1.1 Kreiranje

Proces kreiranja kupona je pojedinačan, odnosno u svakom pozivu funkcije kreira se tačno jedan token. Pored dodele vlasništva nad novim tokenom, funkcija prihvata dodatne parametre koji definišu funkcionalna svojstva kupona.

Mapa vlasnika modifikovana je tako da umesto adrese vlasnika direktno preslikava identifikator tokena u strukturu `CouponInfo`, koja enkodira sve relevantne informacije o kuponu unutar jedne skladišne jedinice, kao što je prethodno opisano. Iz tog razloga funkcija, pored adrese primaoca, prima i početnu vremensku oznaku, indikator prenosivosti, vrednost popusta i broj vremenskih intervala validnosti.

Na početku izvršavanja funkcije vrši se validacija ulaznih parametara kako bi se obezbedila njihova konzistentnost sa definisanim modelom enkodiranja, posebno u pogledu ograničenja reprezentacije procentualne vrednosti i maksimalnog trajanja važenja kupona. Nakon uspešne validacije, strukturirani podaci o kuponu upisuju se u mapu vlasnika na poziciji definisanoj promenljivom `_currentIndex`, nakon čega se ova promenljiva inkrementira.

ID #0	ID #1	ID #2	ID #3	ID #4	ID #5	
vlasnik: Alisa vreme: 01-01-2026 poklonjiv: ✓ procenat: 20 trajanje: 60	vlasnik: Alisa vreme: 01-01-2026 poklonjiv: ✓ procenat: 20 trajanje: 60	vlasnik: Alisa vreme: 01-01-2026 poklonjiv: ✓ procenat: 20 trajanje: 60	vlasnik: Alisa vreme: 01-01-2026 poklonjiv: ✓ procenat: 20 trajanje: 60	vlasnik: Boban vreme: 02-02-2026 poklonjiv: ✗ procenat: 15 trajanje: 90	vlasnik: Boban vreme: 02-02-2026 poklonjiv: ✗ procenat: 15 trajanje: 90	...
SSTORE	SSTORE	SSTORE	SSTORE	SSTORE	SSTORE	

Slika 6.3: Višestruko kreiranje kupona u standardu ERC-721 sa fokusom na trošak skladištenja u mapi vlasništva označen crvenom bojom.

Tokom ove operacije realizuju se sledeće naplate pristupa skladištu:

1. Upis nove skladišne jedinice u mapu vlasnika uzrokuje trošak hladne SSTORE(1) operacije, budući da se radi o prethodno nekorišćenoj skladišnoj jedinici.
2. Ažuriranje mape stanja vlasnika uzrokuje trošak hladne SSTORE(2) operacije, ako nije nula.
3. Promena vrednosti promenljive `_currentIndex` realizuje se kao topla SSTORE(2) operacija, ako nije nula, jer je skladišna jedinica već pristupljena tokom izvršavanja funkcije.

6.3.1.2 Prenos

Na početku izvršavanja funkcije prenosa trenutni vlasnik se učitava u memoriju radi validacije. Nakon toga vrši se provera vlasništva nad tokenom i ispravnosti adrese primaoca. Dodatno, implementacija kupona proširuje osnovni ERC-721 model proverom da li je kupon označen kao prenosiv. Nakon uspešne validacije ažuriraju se brojevi kupona adresa i prenosi vlasništvo.

Tokom ove operacije realizuju se sledeće naplate pristupa skladištu:

1. Učitavanje vlasnika u memoriju radi validacije generiše trošak hladne SLOAD operacije.
2. Stanje pošiljaoca se smanjuje za jedan, što uzrokuje trošak hladne SSTORE(2) operacije, uz mogućnost povraćaja gasa ukoliko nova vrednost postane nula.
3. Stanje primaoca se povećava za jedan, što uzrokuje trošak hladne SSTORE(2) operacije, ako nije nula.
4. Vlasništvo nad tokenom ažurira se izmenom odgovarajuće skladišne jedinice u mapi vlasnika, pri čemu se operacija realizuje uz trošak tople SSTORE(2) operacije, jer je skladišna jedinica već učitana tokom validacije.

ID #0	ID #1	ID #2	ID #3	ID #4	ID #5	
vlasnik: Alisa vreme: 01-01-2026 poklonjiv: ✓ procenat: 20 trajanje: 60	vlasnik: Eva vreme: 01-01-2026 poklonjiv: ✓ procenat: 20 trajanje: 60	vlasnik: Alisa vreme: 01-01-2026 poklonjiv: ✓ procenat: 20 trajanje: 60	vlasnik: Alisa vreme: 01-01-2026 poklonjiv: ✓ procenat: 20 trajanje: 60	vlasnik: Boban vreme: 02-02-2026 poklonjiv: ✗ procenat: 15 trajanje: 90	vlasnik: Boban vreme: 02-02-2026 poklonjiv: ✗ procenat: 15 trajanje: 90	...
SSTORE						

Slika 6.4: Prenos kupona u standardu ERC-721 sa fokusom na trošak skladištenja u mapi vlasništva označen crvenom bojom.

6.3.1.3 Uništavanje

Operacija uništavanja omogućava uklanjanje kupona iz sistema ukoliko period njegove validnosti nije istekao. Kao i kod prenosa, na početku se učitava vlasnik radi validacije. Implementacija kupona uvodi dodatnu proveru zasnovanu na kodiranim vremenskim parametrima, čime se osigurava da je kupon istekao. Ova provera formalno se izražava uslovom:

$$startTimestamp + _BLOCK_DURATION \cdot numOfBlocks \leq block.timestamp \quad (6.1)$$

Ukoliko je uslov ispunjen, skladišna jedinica koja sadrži informacije o kuponu briše se iz mape vlasnika, čime se skladišna lokacija vraća u nulto stanje.

Tokom ove operacije realizuju se sledeće naplate pristupa skladištu:

1. Stanje vlasnika se umanjuje za jednu jedinicu, uz trošak hladne SSTORE(2) operacije i potencijalni dodatni povraćaj gasa ukoliko nova vrednost postane nula.
2. Oslobođanje skladišne jedinice u mapi vlasništva rezultuje povraćajem dela prethodno utrošenog gasa u skladu sa pravilima EVM modela.

ID #0	ID #1	ID #2	ID #3	ID #4	ID #5	
vlasnik: Alisa vreme: 01-01-2026 poklonjiv: ✓ procenat: 20 trajanje: 60	vlasnik: Eva vreme: 01-01-2026 poklonjiv: ✓ procenat: 20 trajanje: 60	0	vlasnik: Alisa vreme: 01-01-2026 poklonjiv: ✓ procenat: 20 trajanje: 60	vlasnik: Boban vreme: 02-02-2026 poklonjiv: ✗ procenat: 15 trajanje: 90	vlasnik: Boban vreme: 02-02-2026 poklonjiv: ✗ procenat: 15 trajanje: 90	...

POVRAĆAJ

Slika 6.5: Uništavanje kupona u standardu ERC-721 sa fokusom na oslobađanje skladišnog prostora u mapi vlasništva označen zelenom bojom.

6.3.2 ERC-721A

U standardu ERC-721, proces kreiranja tokena je strukturalno pojedinačan. Stoga, ukoliko je potrebno kreirati više tokena unutar jedne transakcije, neophodno je koristiti iterativni postupak, pri čemu se mapa vlasništva ažurira u svakoj iteraciji, jer svaki token mora biti evidentiran zasebno [229]. Ovakav pristup dovodi do višestrukog upisivanja identičnih podataka u skladište kada se više međusobno identičnih tokena dodeljuje istoj adresi, što rezultuje značajnim troškovima gasa.

Standard ERC-721A [230] uvodi optimizaciju upravo za ovaj scenario, sa ciljem smanjenja troškova gasa prilikom kreiranja više međusobno ekvivalentnih ERC-721 tokena dodeljenih istoj adresi. Ova optimizacija rešava problem visokih troškova povezanih sa pojedinačnim evidentiranjem vlasništva za svaki token.

ERC-721A omogućava efikasno grupno kreiranje tokena (eng. *batch minting*) tako što se informacija o vlasništvu eksplicitno upisuje samo za prvi token u grupi (eng. *head of the batch*), dok se vlasništvo nad preostalim tokenima implicitno podrazumeva. Na ovaj način vrši se samo jedan upis podataka u skladište pametnog ugovora, čime se ostvaruju uštede gasa tokom kreiranja. Međutim, ova optimizacija uvodi dodatne troškove tokom operacija prenosa i uništavanja tokena, jer implicitno vlasništvo tada mora biti eksplicitno razrešeno.

Standard koristi mapu upakovanih vlasničkih podataka `_packedOwnerships`, koja preslikava identifikator tokena u kompaktnu reprezentaciju vlasništva korišćenjem bitskih operacija. Ova reprezentacija se smešta u jednu skladišnu jedinicu i definiše se kao:

$$owner \mid (timestamp \ll 160) \mid (nextInitialized \ll 225)$$

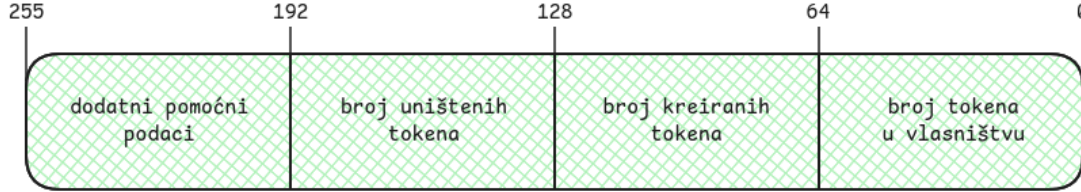
gde:

- *owner* predstavlja adresu vlasnika tokena,
- *timestamp* predstavlja vremensku oznaku kreiranja tokena u *Unix* formatu,
- *nextInitialized* predstavlja binarnu promenljivu koja označava da li su informacije o narednom identifikatoru tokena implicitno definisane.
- Binarna promenljiva koja označava stanje iskorišćenosti (eng. *burned*) implicitno je postavljena na netačno tokom kreiranja.

Pored toga, standard koristi mapu upakovanog stanja podataka `_packedAddressData`, koja preslikava adresu u kompaktnu strukturu podataka od četiri polja po 64 bita, radi praćenja:

- broja tokena u vlasništvu adrese,

- broja kreiranih tokena,
- broja uništenih tokena,
- dodatnih pomoćnih podataka povezanih sa adresom.



Slika 6.6: Reprezentacija informacija stanja adrese enkodiranih u jednu EVM skladišnu jedinicu.

6.3.2.1 Kreiranje

ERC-721A omogućava kreiranje više identičnih tokena unutar jedne transakcije. Funkcija za kreiranje kupona, pored parametara definisanih u osnovnom ERC-721 standardu, dodatno prima i broj kupona koji se kreiraju. Kako su identifikatori tokena uzastopni, pokazivač na naredni identifikator kupona najpre se učitava u memoriju, nakon čega se vrši validacija ulaznih parametara.

Algoritam 6.3 Funkcija kreiranja elektronskih kupona u okviru ERC-721A standarda

```

1: function MINT(to, quantity, startTimestamp, giftable, percentage, numOfBlocks)
2:   startTokenId = _currentIndex
3:   require(quantity > 0)
4:   require(to ≠ address(0))
5:   require(percentage > 0 and percentage ≤ 100)
6:   require(numOfBlocks > 0 and numOfBlocks < 216)
7:   result = to
8:   result |= startTimestamp << 160
9:   result |= (quantity == 1) << 225
10:  result |= giftable << 226
11:  result |= percentage << 232
12:  result |= numOfBlocks << 240
13:  _packedOwnerships[startTokenId] = result
14:  _packedAddressData[to] += quantity · ((1 << 64) | 1)
15:  _currentIndex = startTokenId + quantity
16: end function

```

Informacije o vlasništvu se zatim eksplicitno upisuju samo za prvi token u grupi, korišćenjem sledeće kompaktne reprezentacije:

$$\begin{aligned}
& owner \mid (startTimestamp \ll 160) \mid (nextInitialized \ll 225) \mid \\
& (giftable \ll 226) \mid (percentage \ll 232) \mid (numOfBlocks \ll 240)
\end{aligned} \tag{6.2}$$

Ova reprezentacija proširuje osnovni model tako što umesto vremena kreiranja beleži vreme početka važenja kupona, kao i dodatne informacije specifične za kupon, pri čemu se svi podaci pakuju u jednu skladišnu jedinicu korišćenjem bitskih operacija, umesto korišćenja eksplicitne strukture podataka.

Tokom ove operacije realizuju se sledeće naplate pristupa skladištu:

-
- | ID #0 | ID #1 | ID #2 | ID #3 | ID #4 | ID #5 |
|---|-------|-------|-------|---|-------|
| vlasnik: Alisa
vreme: 01-01-2026
poklonjiv: ✓
uništen: ✗
procenat: 20
trajanje: 60 | 0 | 0 | 0 | vlasnik: Boban
vreme: 02-02-2026
poklonjiv: ✗
uništen: ✗
procenat: 15
trajanje: 90 | 0 |
- Diagram illustrating the SSTORE layout for a table with 6 columns (ID #0 to ID #5). The layout is divided into two SSTORE blocks. The first SSTORE block contains ID #0 (red cross-hatched) and IDs #1, #2, #3 (blue cross-hatched). The second SSTORE block contains ID #4 (red cross-hatched) and ID #5 (blue cross-hatched). IDs #1, #2, #3, and #5 are labeled "implicitno definisano". ID #0 contains data for Alisa, and ID #4 contains data for Boban. IDs #1, #2, #3, and #5 contain the value 0. Ellipses (..) follow ID #5.

Ključna optimizacija proizilazi iz činjenice da se operacija `SSTORE(1)` izvršava samo jednom, za prvi token u grupi, dok se za ostale tokene vlasništvo implicitno podrazumeva, čime se značajno smanjuju ukupni troškovi gasa.

Operacija prenosa kupona zadržava istu funkcionalnu logiku kao u ERC-721 modelu, uz dodatni korak određivanja eksplicitnih informacija o vlasništvu. Budući da vlasništvo može biti implicitno, pre validacije prenosa vrši se pronalaženje odgovarajuće inicijalizovane skladišne jedinice pretragom unazad kroz mapu vlasništva, počevši od identifikatora datog kupona, nakon čega se pronađena vrednost smešta u memoriju radi korišćenja tokom izvršavanja funkcije.

$$startTokenId() \leq tokenId \ \&\& \ tokenId < \ currentIndex \quad (6.3)$$

103

Algoritam 6.4 Algoritam za pronalaženje implicitnog vlasništva u ERC-721A standardu

```
1: function _PACKEDOWNERSHIPOf(id)
2:   if _startTokenId() > id || id ≥ _currentIndex then                                cold SLOAD
3:     revert()
4:   end if
5:   packed = _packedOwnerships[id]                                                    cold SLOAD
6:   if packed ≠ 0 then
7:     return packed
8:   end if
9:   while true do
10:    packed = _packedOwnerships[--id]                                                  cold SLOAD
11:    if packed ≠ 0 then
12:      if packed & (1 << 224) == 0 then
13:        return packed
14:      else
15:        revert()
16:      end if
17:    end if
18:  end while
19: end function
```

Dodatno, pre vraćanja rezultata, u okviru iste funkcije proverava se da li je token iskorišćen, proverom binarne promenljive koja označava stanje iskorišćenosti tokena.

Ova operacija ima iterativnu prirodu, i u svakoj iteraciji vrši se učitavanje prethodne skladišne jedinice i poređenje njene vrednosti sa nulom, kako bi se utvrdilo da li je pronađen prvi token u grupi. Zbog toga trošak ove operacije raste proporcionalno rastojanju između posmatranog tokena i prvog tokena u grupi. Ako $n \geq 1$ predstavlja ovo rastojanje, procenjeni trošak funkcije može se aproksimirati kao:

$$(n + 2) \cdot \text{SLOAD} + (n + 4) \cdot 141 \quad (6.4)$$

gde prvi član predstavlja trošak operacija učitavanja iz skladišta, dok drugi član predstavlja aproksimaciju troška ostalih instrukcija. Instrukcije van petlje su zaokružene tako da koštaju četiri puta više od instrukcija unutar petlje, radi kompaktnije formule. Procena je zasnovana na empirijskoj analizi višestrukih pokretanja algoritama, sa širokim rasponom udaljenosti između početka serije i traženog identifikatora tokena iz istog grupnog kreiranja.

Nakon pronalaženja vlasnika, ažuriraju se odgovarajući podaci o vlasništvu i stanju adresa. Ukoliko se prenosi token čije je vlasništvo implicitno definisano, potrebno je eksplicitno inicijalizovati skladišnu jedinicu za taj token, čime se implicitno grupno vlasništvo deli na dva segmenta, koje je zatim potrebno eksplicitno evidentirati. Ovo predstavlja dodatni trošak u odnosu na standard ERC-721.

Tokom ove operacije realizuju se sledeće naplate pristupa skladištu:

1. Trošak određivanja implicitnog vlasništva, izračunat formulom 6.4.
2. Smanjenje stanja pošiljaoca, što uzrokuje trošak hladne operacije `SSTORE(2)`.
3. Povećanje stanja primaoca, što uzrokuje trošak hladne operacije `SSTORE(2)`, ako nije nula.
4. Ažuriranje informacija o vlasništvu za preneti token, što uzrokuje trošak tople operacije `SSTORE(2)`.

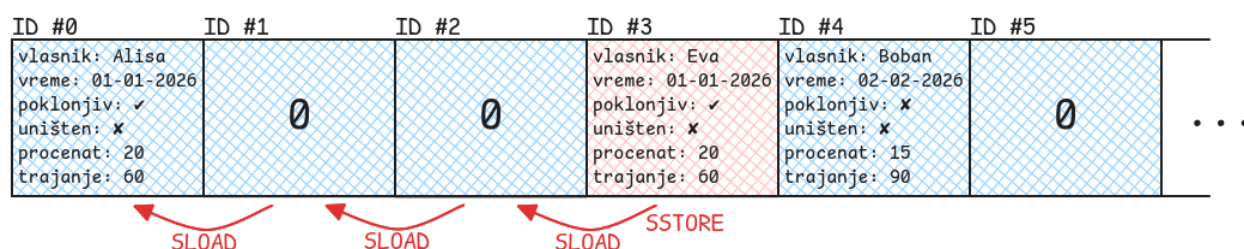
5. Potencijalni dodatni upis u narednu skladišnu jedinicu radi očuvanja implicitnog vlasništva preostalih tokena u grupi, što uzrokuje trošak hladne operacije SSTORE(1).

Algoritam 6.5 Funkcija prenosa elektronskih kupona u okviru ERC-721A standarda

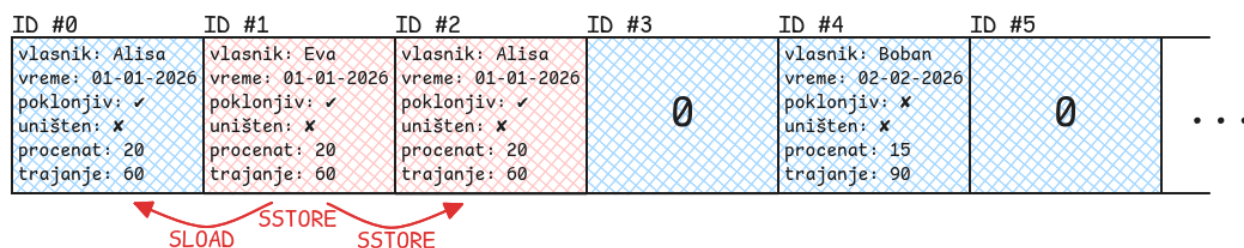
```

1: function TRANSFERFROM(from,to,tokenId)
2:   prevOwnershipPacked = _packedOwnerships[tokenId]
3:   require(address(uint160(prevOwnershipPacked)) == from)
4:   require(to  $\neq$  address(0))
5:   require(owners[id].giftable == true)
6:   _packedAddressData[from]--
7:   _packedAddressData[to]++
8:   result = to
9:   result |= 1  $\ll$  225
10:  result |= (prevOwnershipPacked  $\gg$  160)  $\ll$  160
11:  _packedOwnerships[tokenId] = result
12:  if prevOwnershipPacked & (1  $\ll$  225) == 0 then
13:    nextTokenId = tokenId + 1
14:    if _packedOwnerships[nextTokenId] == 0 then
15:      if nextTokenId  $\neq$  _currentIndex then
16:        _packedOwnerships[nextTokenId] = prevOwnershipPacked
17:      end if
18:    end if
19:  end if
20: end function

```



(a) Ažuriranje vlasništva u slučaju kada se implicitno vlasništvo ne prekida.



(b) Ažuriranje vlasništva uz dodatni upis u sledeću skladišnu jedinicu radi očuvanja implicitnog vlasništva.

Slika 6.8: Prenos kupona u standardu ERC-721A sa fokusom na trošak skladištenja u mapu vlasništva označen crvenom bojom.

U implementaciji e-kupona, vremenska oznaka kreiranja se ne ažurira tokom prenosa, kako bi se očuvala informacija o vremenu početka važenja kupona.

6.3.2.3 Uništavanje

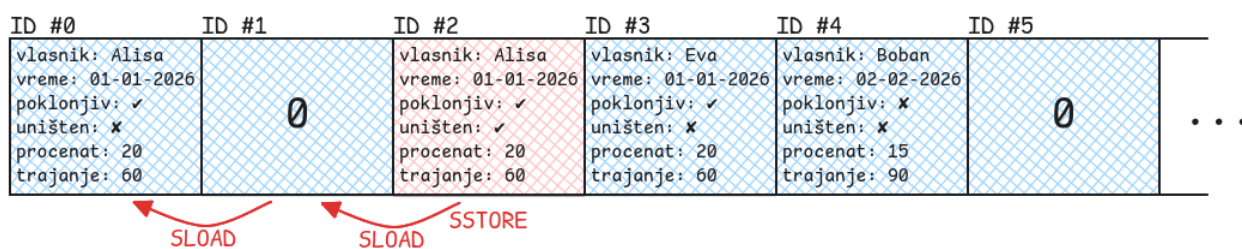
Operacija uništavanja prati sličan postupak kao prenos. Najpre se pronalazi vlasnik tokena korišćenjem iste metode za određivanje implicitnog vlasništva, nakon čega se ažuriraju odgovarajući podaci o stanju adrese i vlasništvu nad tokenom.

Važno je napomenuti da, iako token postaje trajno nevažeći, odgovarajuća skladišna jedinica ostaje inicijalizovana. Kao posledica toga, ne dolazi do povraćaja gasa, što predstavlja dodatni nedostatak u odnosu na standard ERC-721.

Tokom ove operacije realizuju se sledeće naplate pristupa skladištu:

1. Trošak određivanja implicitnog vlasništva, izračunat formulom 6.4.
2. Ažuriranje stanja vlasnika, što uzrokuje trošak hladne operacije $SSTORE(2)$.
3. Ažuriranje informacija o vlasništvu za uništeni token, što uzrokuje trošak tople operacije $SSTORE(2)$.
4. Potencijalni dodatni upis radi očuvanja implicitnog vlasništva preostalih tokena u grupi, što uzrokuje trošak hladne operacije $SSTORE(1)$.

Vremenom, kako se implicitno definisani tokeni prenose ili uništavaju, broj implicitno definisanih vlasništava se smanjuje, a trošak budućih operacija se približava trošku standardne $SSTORE$ operacije, čime se početne uštede gasa postepeno amortizuju.



(a) Uništavanje tokena u slučaju kada se implicitno vlasništvo ne prekida.



(b) Uništavanje tokena koje zahteva dodatni upis u sledeću skladišnu jedinicu radi očuvanja implicitnog vlasništva.

Slika 6.9: Uništavanje kupona u standardu ERC-721A sa fokusom na trošak skladištenja u mapi vlasništva označen crvenom bojom.

6.3.3 ERC-721Psi

Standard ERC-721Psi [231] predstavlja alternativu standardu ERC-721A sa dodatnim mehanizmom koji rešava problem neefikasne pretrage prvog tokena u grupi. Na taj način se otklanja ključni

nedostatak implicitnog modela vlasništva u standardu ERC-721A, posebno u opštem slučaju kada je veličina grupe novokreiranih tokena proizvoljno velika.

Radi postizanja ove efikasnosti, standard uvodi sledeće komponente:

1. Dodatnu mapu u obliku bitmapi *_batchHead*, koja efikasno beleži početak svake grupe postavljanjem jednog bita na poziciji odgovarajućeg identifikatora.
2. De Bruijnov ciklus [232] reda P_8 , koji omogućava efikasno određivanje pozicije prvog postavljenog bita u bitmapi,
3. Pretraživačku heš-tabelu koja, na osnovu vrednosti dobijene De Bruijnovim preslikavanjem, omogućava direktno određivanje indeksa početnog tokena u grupi.

Uvođenje bitmape uvodi dodatne troškove pri izvršavanju osnovnih funkcija standarda, jer je potrebno održavati njeno stanje tokom svih pomenutih operacija. Međutim, De Bruijnov ciklus i odgovarajuća pretraživačka tabela predstavljaju konstante poznate u vreme kompajliranja te su uključene u bajtkod za izvršavanje pametnog ugovora. Njihovo korišćenje podrazumeva isključivo pristup virtuelnoj memoriji samo za čitanje, bez pristupa skladištu pametnog ugovora, te stoga ne stvara dodatne troškove u pogledu potrošnje gasa.

Za razliku od ranije opisanih standarda, osnovna implementacija ERC-721Psi ne uključuje praćenje stanja adresa niti funkcionalnost uništavanja tokena. Ove mogućnosti se uvode kroz sledeća proširenja pametnog ugovora:

1. *ERC721PsiAddressData* definiše strukturu podataka ekvivalentnu strukturi iz standarda ERC-721A, korišćenjem mape upakovanog stanja adresa *_packedAddressData*. Ovo proširenje eksplicitno razdvaja funkcionalnost praćenja stanja adresa od osnovne implementacije standarda, čime se omogućava modularno uključivanje ove funkcionalnosti samo kada je potrebna.
2. *ERC721PsiBurnable* uvodi dodatnu bitmapu *_burnedToken*, koja beleži stanje uništenosti tokena. Za efikasno određivanje relevantnih pozicija u bitmapi koristi se isti De Bruijnov ciklus i odgovarajuća pretraživačka heš-tabela kao i u slučaju originalne bitmape *_batchHead*.

Navedena proširenja su uključena radi usklađivanja eksperimentalnih uslova sa prethodnim standardima, kako bi se omogućilo konzistentno i uporedivo merenje potrošnje gasa.

6.3.3.1 Kreiranje

Proces kreiranja kupona prima iste ulazne argumente kao odgovarajuća funkcija u standardu ERC-721A. Na početku izvršavanja pokazivač na naredni identifikator tokena učitava se iz skladišta u memoriju, nakon čega se sprovede iste validacije ulaznih parametara.

Informacije o vlasništvu se zatim eksplicitno upisuju samo za prvi token u grupi, korišćenjem formule 6.2, gde je binarna promenljiva *nextInitialized* izostavljena, jer se u ovom standardu ne koristi. Time se omogućava proširenje dostupnog prostora za jedan bit unutar upakovane strukture vlasničkih podataka u odnosu na standard ERC-721A.

Tokom kreiranja neophodno je postaviti odgovarajući bit u bitmapi *_batchHead*, kojim se označava prvi token u grupi. Kako skladišna jedinica u EVM-u ima veličinu od 256 bita, bitmapa je organizovana kao mapa uzastopnih blokova (eng. *bucket*) od po 256 bita. U ovoj reprezentaciji, vrednost 1 na određenoj poziciji označava da identifikator tokena koji odgovara toj poziciji predstavlja početak grupe.

Postupak postavljanja odgovarajućeg bita sastoji se od određivanja indeksa bloka kojem pripada dati identifikator, nakon čega se postavlja odgovarajući bit unutar tog bloka, kao što je prikazano u Algoritmu 6.6.

Algoritam 6.6 Postavljanje prvog u grupi u bitmapi standarda ERC-721Psi

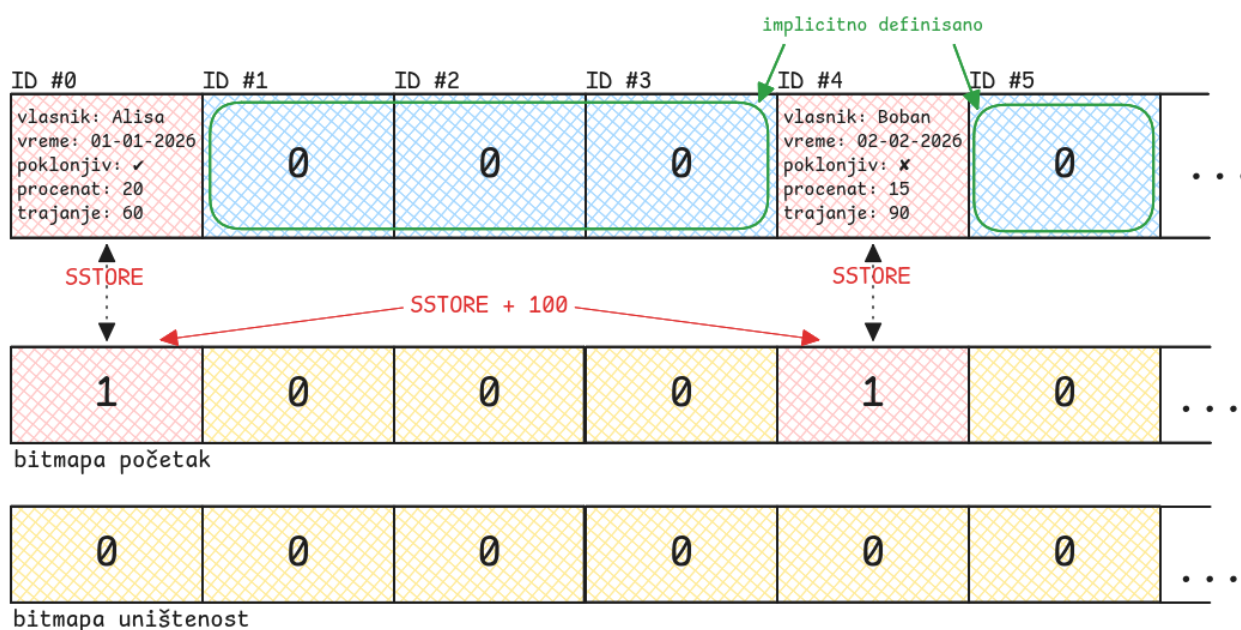
```

1: function BITMAP_SET(id)
2:   bucket = id >> 8
3:   mask = (1 << 255) >> (id & 0xff)
4:   bitmap[bucket] |= mask
5: end function

```

Tokom ove operacije realizuju se sledeće naplate pristupa skladištu:

1. Učitavanje vrednosti pokazivača na naredni identifikator tokena uzrokuje trošak hladne operacije SLOAD.
2. Upis vlasničkih podataka za prvi identifikator u grupi uzrokuje trošak hladne operacije SSTORE(1).
3. Ažuriranje bitmape na odgovarajućoj poziciji uzrokuje trošak hladne operacije SSTORE(2) ako dati blok nije nula.
4. Ažuriranje stanja primaoca uzrokuje trošak hladne operacije SSTORE(2), ako nije nula.
5. Ažuriranje vrednosti pokazivača na naredni identifikator realizuje se kao topla SSTORE(2) operacija, ako nije nula, jer je skladišna jedinica već pristupljena tokom validacije funkcije.



Slika 6.10: Višestruko kreiranje kupona u standardu ERC-721Psi sa fokusom na implicitno vlasništvo označeno zelenom bojom, omogućeno pomoću bitmape početaka, i trošak skladištenja označen crvenom bojom.

6.3.3.2 Prenos

Operacija prenosa kupona zadržava istu osnovnu funkcionalnu logiku kao u standardu ERC-721A, uz dodatni korak ažuriranja bitmape. Određivanje vlasnika tokena zasniva se na pronalaženju identifikatora prvog tokena u grupi označenog u bitmapi, nakon čega se na osnovu te informacije očitava vlasništvo tokena iz mape upakovanih vlasničkih podataka.

Pre početka postupka razrešavanja implicitnog vlasništva, vrši se validacija identifikatora tokena prosleđenog funkciji, pri čemu je ovaj korak, za razliku od prethodnog standarda, izdvojen iz funkcije za pronalaženje implicitnog vlasništva. Ovaj korak uključuje proveru da li identifikator pripada važećem opsegu definisanom uslovom 6.3, kao i dodatnu proveru da li je token označen kao uništen u bitmapi `_burnedToken`. Nakon uspešne validacije, određivanje vlasništva realizuje se korišćenjem funkcije `_getBatchHead`, koja pronalazi identifikator prvog tokena u odgovarajućoj grupi.

Za razliku od standarda ERC-721A, gde se pretraga realizuje sekvencijalnim proveravanjem pojedinačnih identifikatora, ERC-721Psi koristi bitmapu `_batchHead`, čime se omogućava paralelnu obradu 256 identifikatora. Na ovaj način se broj potrebnih iteracija značajno smanjuje, što dovodi do znatno efikasnijeg određivanja implicitnog vlasništva.

Algoritam 6.7 za pronalaženje prvog tokena u grupi sastoji se iz sledećih koraka:

1. Određuje se indeks skladišnog bloka u bitmapi kome pripada posmatrani identifikator, kao i njegova relativna pozicija unutar tog bloka.
2. Pronađeni identifikator u datom bloku pomera se na krajnju desnu poziciju unutar bloka bitmape primenom bitovskog pomeranja udesno. Ovim postupkom razmak između prvog tokena u grupi i prosleđenog identifikatora postaje ekvivalentan poziciji najmanje značajnog postavljenog bita (eng. *least significant one bit*, *LS1B*), za koji je poznat efikasan algoritam [233], razvijen za procesore koji ne poseduju instrukcije za brojanje vodećih ili završnih nula.
3. Primenom De Bruijnovog preslikavanja i odgovarajuće pretraživačke heš-tabele, koja je ugrađena u bajtkod pametnog ugovora, određuje se pozicija najmanje značajnog postavljenog bita.
4. Na osnovu dobijene pozicije rekonstruiše se pozicija u bitmapi identifikatora prvog tokena u grupi, pomoću koje može da se odredi skladišna jedinica koja sadrži eksplicitno definisane informacije o vlasništvu.

Važno je napomenuti da tokom izvršavanja Koraka 2 može doći do slučaja da je vrednost posmatranog bloka jednaka nuli, što znači da se prvi token u grupi nalazi u nekom od prethodnih blokova. U tom slučaju se u petlji učitavaju prethodni blokovi sve dok se ne pronađe blok sa nenultom vrednošću, koji po konstrukciji sadrži identifikator prvog tokena u odgovarajućoj grupi.

Izračunavanje indeksa LS1B unutar jednog bloka ima konstantnu vremensku složenost, nezavisno od pozicije LS1B u okviru bloka. Time se omogućava istovremena obrada 256 lokacija uz konstantan trošak.

Ako sa $n \geq 1$ označimo rastojanje između posmatranog tokena i prvog tokena u grupi, procenjeni trošak funkcije opisane Algoritmom 6.7 može se aproksimirati izrazom

$$\left(\frac{n}{256} + 1\right) \cdot \text{SLOAD} + \left(\frac{n}{256} + 2.5\right) \cdot 194 \quad (6.5)$$

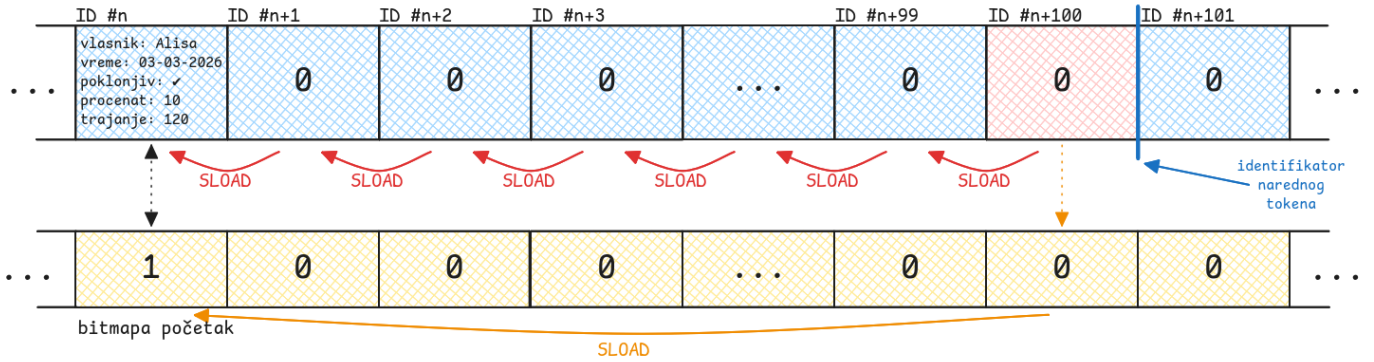
gde prvi član, kao i ranije, predstavlja trošak operacija učitavanja iz skladišta, dok drugi član predstavlja aproksimaciju svih preostalih instrukcija. Cena tih instrukcija je zaokružena radi kompaktnije formulacije, pri čemu je procena zasnovana na istoj empirijskoj analizi kao i u prethodnom slučaju.

Algoritam 6.7 Algoritam za pronalaženje implicitnog vlasništva u ERC-721Psi standardu

```
1: function _GETBATCHHEAD(id)
2:   bucket = id  $\gg$  8
3:   bucketIndex = (id & 0xff)
4:   bb = bitmap[bucket] cold SLOAD
5:   bb = bb  $\gg$  (255 - bucketIndex)
6:   if bb > 0 then
7:     return (bucket  $\ll$  8) | (bucketIndex - DeBrujin(bb))
8:   else
9:     while true do
10:      bucket--
11:      bb = bitmap[bucket] cold SLOAD
12:      if bb > 0 then
13:        return (bucket  $\ll$  8) | (255 - DeBrujin(bb))
14:      end if
15:    end while
16:  end if
17: end function
```

Upravo zbog učitavanja 256 lokacija istovremeno, ova metoda je značajno jeftinija od sekvencijalne pretrage. Petlja za pronalaženje prvog tokena u grupi pomoću bitmape ponavlja se znatno približno ređe nego u ERC-721A standardu, uprkos nešto većem broju aritmetičkih i bitovskih operacija.

Na primer, za $n = 1000$ procenjeni trošak je približno ≈ 134 puta manji u odnosu na ERC-721A, dok je za $n = 10000$ trošak približno ≈ 225 puta manji. U graničnom slučaju kada $n \rightarrow \infty$, odnos troškova teži faktoru 256.



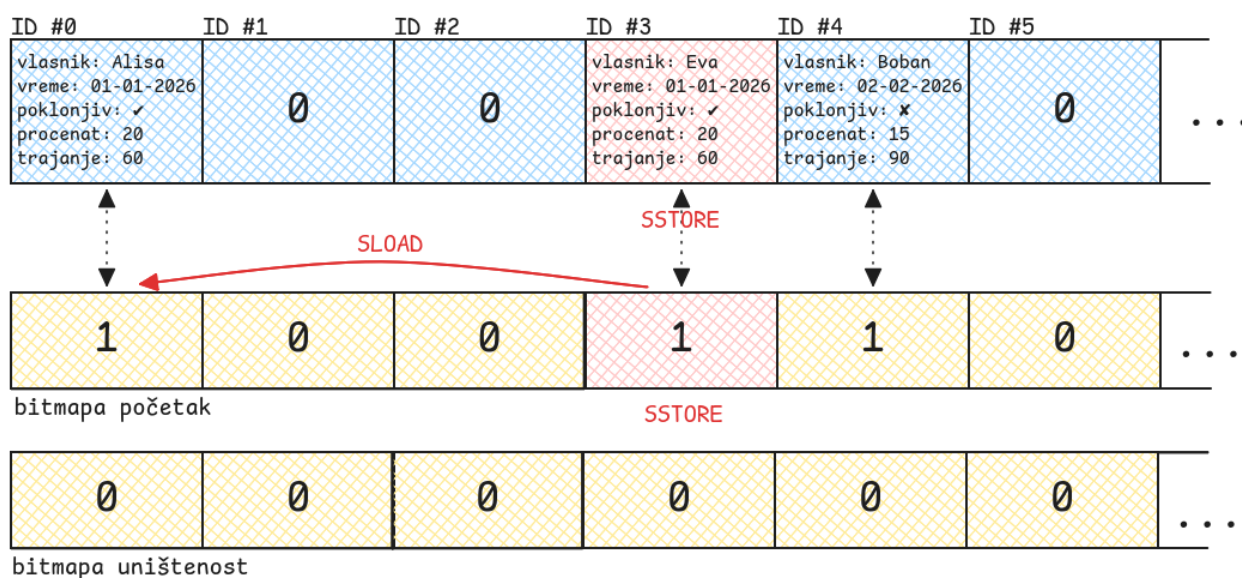
Slika 6.11: Poređenje skladišnih operacija za čitanje prilikom razrešavanja implicitnog vlasništva u standardima ERC-721A i ERC-721Psi, označenih crvenom i narandžastom bojom.

Nakon pronalaženja implicitnog vlasništva, analogno prethodnom standardu, grupno vlasništvo se deli na dva segmenta, nakon čega je potrebno eksplicitno evidentirati ovu promenu u bitmapi. Nakon toga se ažuriraju odgovarajući podaci o vlasništvu i stanju adresa kao i u prethodnom standardu.

Tokom ove operacije realizuju se sledeće naplate pristupa skladištu:

1. Provera da li dati identifikator postoji, sa dodatnom proverom da li je token označen kao uništen, uz trošak dve hladne SLOAD operacije.

2. Trošak određivanja pozicije implicitnog vlasništva u bitmapi, izračunat formulom 6.5.
3. Učitavanje iz mape upakovanog stanja podataka na osnovu pronađene pozicije implicitnog vlasništva u bitmapi, što uzrokuje trošak hladne SLOAD operacije.
4. Ažuriranje informacija o vlasništvu za preneti token, što uzrokuje trošak hladne operacije SSTORE(1).
5. Ažuriranje bitmape za preneti token, sa troškom od 100 gasa jer je skladišna jedinica već pristupljena, ukoliko nije prethodno pristupljena toplom SSTORE(2) operacijom.
6. Potencijalno ažuriranje mape upakovanog stanja podataka radi očuvanja implicitnog vlasništva, što uzrokuje trošak hladne operacije SSTORE(1).
7. Potencijalno ažuriranje bitmape originalnog vlasnika, što uzrokuje trošak tople operacije SSTORE(2) ako naredni bitmap blok nije prazan, odnosno tople operacije SSTORE(1) ako je prazan.



Slika 6.12: Prenos kupona u standardu ERC-721Psi u slučaju kada se implicitno vlasništvo ne prekida, sa fokusom na trošak skladištenja u bitmapi označen crvenom bojom.

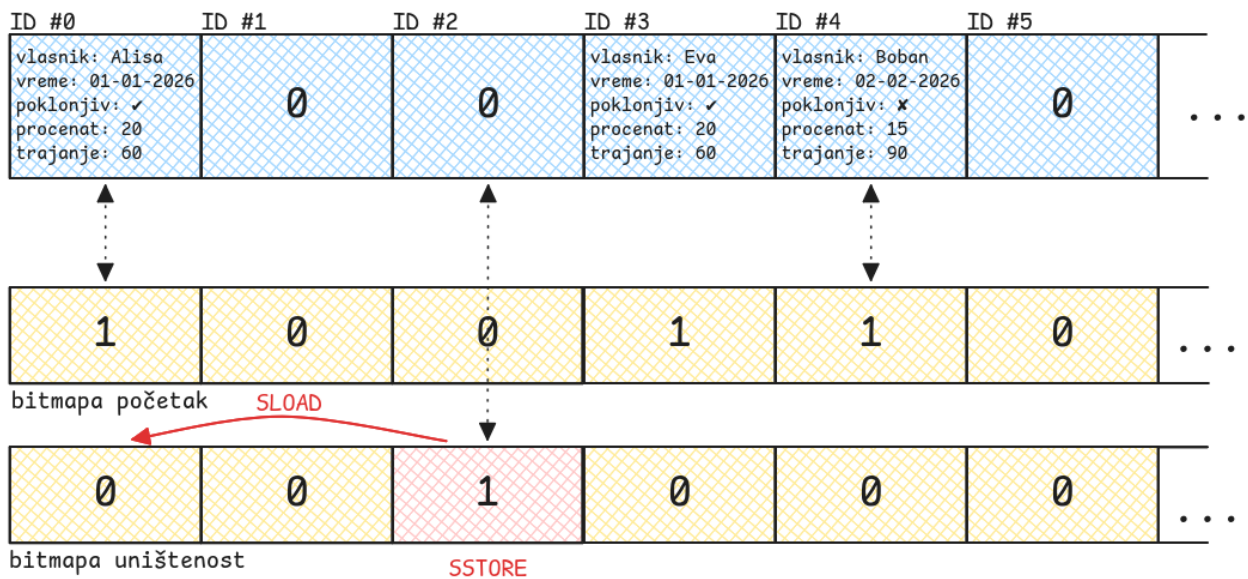
6.3.3.3 Uništavanje

Operacija uništavanja prati istu osnovnu logiku kao i u ERC-721A standardu, uz dodatno ažuriranje bitmape koja evidentira stanje uništenosti tokena. Najpre se određuje implicitni vlasnik tokena pretraživanjem bitmape početaka grupa, čime se istovremeno implicitno proverava da li je kupon već iskorišten. Nakon određivanja informacija o vlasništvu, proverava se da li su ispunjeni uslovi za uništavanje primenom formule 6.1. Ukoliko je provera uspešna, evidentira se stanje uništenosti tokena i ažurira odgovarajuće stanje u mapi upakovanog stanja adresa.

Zbog korišćenja ekstenzije koja beleži stanje uništenosti tokena, ova operacija podrazumeva postavljanje odgovarajućeg bita u bitmapi uništenosti na vrednost 1. Kao posledica ovakvog pristupa, iako token postaje trajno nevažeći, ne dolazi do povraćaja gasa, jer se tokom uništavanja vrši dodatni upis u skladište, analogno prethodnom standardu.

Tokom ove operacije realizuju se sledeće naplate pristupa skladištu:

1. Provera da li dati identifikator postoji, sa dodatnom proverom da li je token označen kao uništen, uz trošak dve hladne SLOAD operacije.
2. Trošak određivanja pozicije implicitnog vlasništva u bitmapi, izračunat formulom 6.5.
3. Učitavanje iz mape upakovanog stanja podataka na osnovu pronađene pozicije implicitnog vlasništva u bitmapi, što uzrokuje trošak hladne SLOAD operacije.
4. Ažuriranje bitmape uništenosti tokena, što uzrokuje trošak tople SSTORE(2) operacije, ako nije nula.
5. Ažuriranje stanja korisnika u mapi upakovanog stanja adresa, što uzrokuje trošak hladne SSTORE(2) operacije, ako nije nula.



Slika 6.13: Uništavanje kupona u standardu ERC-721Psi sa fokusom na trošak skladištenja u bitmapi uništenih tokena označen crvenom bojom.

6.4 Rezultati eksperimentalne analize

Nakon definisanja različitih pristupa implementaciji elektronskih kupona kroz NFT standarde, kao i analize njihovih operacija, za eksperimentalno određivanje potrošnje gasa korišćen je razvojni okvir *Foundry* [234], koji predstavlja široko prihvaćen alat za razvoj i testiranje pametnih ugovora. Ovaj okvir obuhvata alat *forge* (v0.2.0), koji se koristi za izgradnju, testiranje i implementaciju pametnih ugovora, kao i alat *anvil* (v0.2.0), koji obezbeđuje lokalni čvor Eterijum testne mreže. U okviru ovog okruženja implementirani su pametni ugovori za sve posmatrane standarde, pri čemu je merena potrošnja gasa za izvršavanje operacija nad elektronskim kuponima.

Radi obezbeđivanja doslednosti i uporedivosti rezultata, svi pametni ugovori kompajlirani su korišćenjem kompajlera *solc* v0.8.22, sa onemogućenom optimizacijom. Za implementaciju ERC-721

standarda korišćena je referentna implementacija biblioteke *OpenZeppelin* v4.9.0 [235], dok su za standarde ERC-721A i ERC-721Psi korišćene implementacije v4.2.3, odnosno v0.7.0.

U cilju preciznog poređenja efikasnosti implementacija, analizirana je isključivo potrošnja gasa posmatranih operacija nad kuponima, bez uključivanja osnovnog troška transakcije od 21.000 jedinica gasa, koji je konstantan i nezavisan od implementacije standarda. Pre sprovođenja eksperimenata izvršena je inicijalizacija svih relevantnih internih promenljivih stanja, uključujući promenljivu *_currentIndex* kod svih standarda, promenljivu *_burnCounter* kod ERC-721A standarda, kao i obe bitmape kod ERC-721Psi standarda.

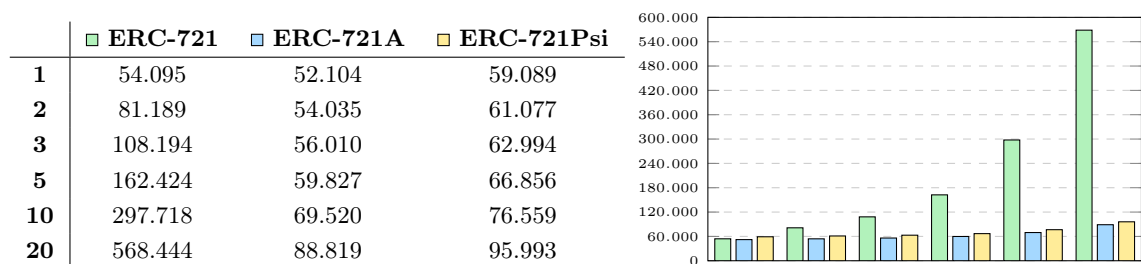
U eksperimentalnom delu analizirana je zavisnost potrošnje gasa od broja kupona kreiranih u jednoj transakciji. Testovi su sprovedeni pozivanjem funkcije za kreiranje sa vrednostima parametra $n \in \{1, 2, 3, 5, 10, 20\}$, pri čemu je za svaku vrednost izmerena ukupna potrošnja gasa. Važno je napomenuti da je broj kreiranih kupona jedini parametar koji utiče na potrošnju gasa ove operacije, dok ostali parametri imaju konstantan uticaj.

Analogno tome, za analizu operacija prenosa i uništavanja korišćen je identifikator kupona oblika $n + 1$, koji predstavlja rastojanje između prvog tokena u grupi i posmatranog tokena. Na ovaj način simuliran je najgori slučaj određivanja implicitnog vlasništva u okviru grupe. U slučaju analize uništavanja više kupona u jednoj transakciji, funkcija uništavanja pozivana je uzastopno za različite identifikatore kupona, čime je obezbeđeno da se u svakom pozivu obrađuje različiti kupon.

Kreiranje više kupona u jednoj transakciji

Pretpostavlja se da adrese korisnika imaju nulta početna stanja pre operacije kreiranja, što odgovara scenariju u kome korisnik po prvi put koristi platformu. U tom slučaju, trošak kreiranja prvog kupona veći je za 17.100 jedinica gasa u odnosu na naredna kreiranja, što odgovara razlici između operacija *SSTORE(1)* i *SSTORE(2)*. Ova razlika proizlazi iz činjenice da se pri prvom upisu u mapu stanja vrši prelaz iz nulte u nenultu vrednost, dok se kod narednih upisa ažurira već inicijalizovana lokacija.

Kao što je očekivano, standard ERC-721 ostvaruje najveće troškove gasa u poređenju sa standardima ERC-721A i ERC-721Psi, jer podrazumeva eksplicitno čuvanje informacija o vlasništvu za svaki pojedinačni kupon. Nasuprot tome, standardi ERC-721A i ERC-721Psi postižu značajno niže troškove zahvaljujući implicitnom modelovanju vlasništva u okviru sekvencijalno dodeljenih identifikatora, pri čemu je ERC-721Psi neznatno skuplji zbog dodatnog troška ažuriranja bitmape.



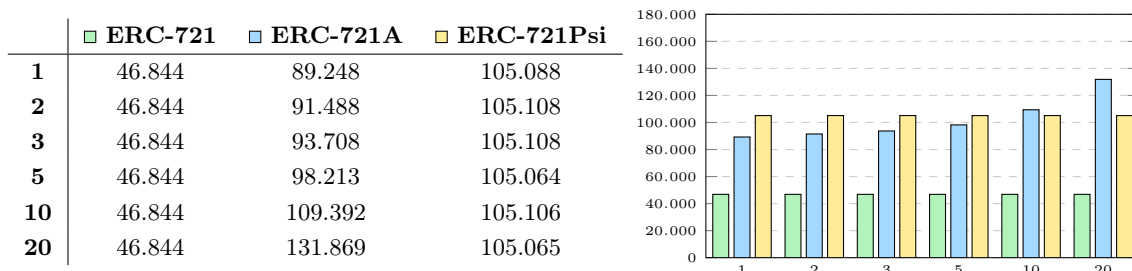
Rezultat 1: Potrošnja gasa različitih NFT standarda prilikom višestrukog kreiranja kupona u jednoj transakciji

Prenos kupona u zavisnosti od udaljenosti od početka grupe

Pretpostavlja se da adresa primaoca ima nulto početno stanje, dok adresa pošiljaoca poseduje više kupona, što predstavlja tipičan slučaj u praktičnoj primeni. Kao što je očekivano, trošak prenosa kod standarda ERC-721 ostaje konstantan, budući da se vlasništvo nad svakim kuponom eksplicitno čuva nezavisno od ostalih kupona.

Za standard ERC-721A uočava se da je trošak prenosa manji za kupone koji su bliže početku grupe, ali raste sa povećanjem udaljenosti, što je posledica potrebe za pronalaženjem implicitno definisanog vlasništva. Suprotno tome, standard ERC-721Psi ostvaruje približno konstantan trošak prenosa, jer se informacija o početku grupe efikasno pronalazi korišćenjem bitmape, pri čemu se, za posmatrane parametre testiranja, svi kuponi nalaze unutar istog bloka bitmape.

Dalja analiza pokazuje da, analogno operaciji kreiranja, trošak prenosa opada za 17.100 jedinica gasa ukoliko adresa primaoca već poseduje nenultu vrednost u mapi stanja. Takođe, kod standarda ERC-721A i ERC-721Psi, ukoliko preneti kupon predstavlja početak grupe ili, u slučaju ERC-721A, ima postavljenu binarnu promenljivu *_nextInitialized*, izbegava se dodatna *SSTORE(1)* operacija, čime se ostvaruje dodatno smanjenje troškova. Konačno, jedino standard ERC-721 ostvaruje povraćaj gasa u slučaju kada se nakon prenosa stanje pošiljaoca smanjuje na nulu, što rezultuje smanjenjem ukupnog troška prenosa za 4.800 jedinica gasa.



Rezultat 2: Potrošnja gasa različitih NFT standarda prilikom prenosa kupona u zavisnosti od udaljenosti od početnog elementa grupe

Uništavanje kupona u zavisnosti od udaljenosti od početka grupe

Pretpostavlja se da korisnik nakon uništavanja kupona zadržava nenulto stanje, što predstavlja realističan scenario imajući u vidu da se analiza odnosi na slučaj grupnog kreiranja više kupona u okviru jedne transakcije.

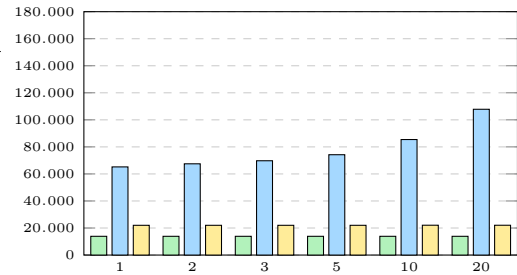
Trošak uništavanja kod standarda ERC-721 ostaje konstantan, nezavisno od udaljenosti kupona unutar grupe, budući da se vlasništvo nad svakim kuponom čuva eksplicitno i nezavisno. Analogno operaciji prenosa, povraćaj gasa primenjuje se samo kod ovog standarda, što ga čini troškovno najpovoljnijim.

Kod standarda ERC-721Psi trošak uništavanja ostaje približno konstantan pod pretpostavkom da se svi posmatrani kuponi nalaze unutar istog bloka bitmape. U tom slučaju, ažurira se već inicijalizovani blok, čija je vrednost nenulta, što je realistična pretpostavka imajući u vidu da jedan blok bitmape obuhvata 256 uzastopnih identifikatora.

Sa druge strane, standard ERC-721A ostvaruje najveće troškove, posebno pri većim udaljenostima od početka grupe, usled potrebe za pronalaženjem implicitno definisanog vlasništva. Dodatno, operacija uništavanja dovodi do razdvajanja prethodno jedinstvene grupe na dva

segmenta, pri čemu je neophodno eksplicitno evidentirati granice novonastalih segmenata, što generiše dodatne troškove.

	ERC-721	ERC-721A	ERC-721Psi
1	13.873	65.199	22.006
2	13.873	67.485	22.007
3	13.873	69.745	22.008
5	13.873	74.209	22.006
10	13.873	85.435	22.072
20	13.873	107.822	22.050



Rezultat 3: Potrošnja gasa različitih NFT standarda prilikom uništavanja kupona u zavisnosti od udaljenosti od početnog elementa grupe

Uništavanje više kupona u jednoj transakciji

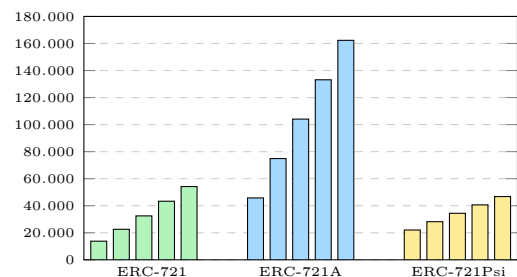
Korisnicima je omogućeno da unište više kupona u okviru jedne transakcije. U ovom eksperimentu pretpostavlja se da proces uništavanja započinje od prvog kupona u sekvenci, pri čemu se broj uzastopno uništenih kupona varira za $n \in \{1, 2, 3, 4, 5\}$. Cilj ove analize jeste da se prikaže opšte ponašanje pri višestrukome uništavanju kupona u okviru iste transakcije, bez razmatranja slučajeva u kojima se kuponi nalaze na proizvoljnim pozicijama unutar prethodno formirane grupe.

Uočava se da standard ERC-721 ostvaruje najniži početni trošak, budući da se prilikom uništavanja oslobađa skladišna jedinica i ostvaruje povraćaj gasa od 4.800 jedinica. Međutim, svaka naredna operacija podrazumeva dodatne skladišne pristupe, pri čemu se i dalje plaća razlika između troška skladišnog pristupa i ostvarivog povraćaja, tako da ukupni trošak raste približno linearno sa brojem uništenih kupona.

Kod standarda ERC-721Psi ne postoji mehanizam povraćaja gasa, ali se tokom uzastopnih uništavanja ažurira ista bitmap struktura koja evidentira stanje kupona. Nakon prvog upisa, svaka naredna izmena unutar iste transakcije predstavlja skladišni pristup, čiji je trošak 100 jedinica gasa. Posledično, počevši od određenog broja uništenih kupona, standard ERC-721Psi postaje troškovno povoljniji u odnosu na ERC-721.

Standard ERC-721A ostvaruje najveće troškove u ovom scenariju, budući da se za svaki uništen kupon eksplicitno ažuriraju informacije o vlasništvu, bez ostvarivanja dodatnog povraćaja gasa, što dovodi do rasta ukupne potrošnje.

	ERC-721	ERC-721A	ERC-721Psi
1	13.786	45.770	22.051
2	22.580	74.880	28.196
3	32.493	104.079	34.412
4	43.332	133.149	40.627
5	54.191	162.327	46.773

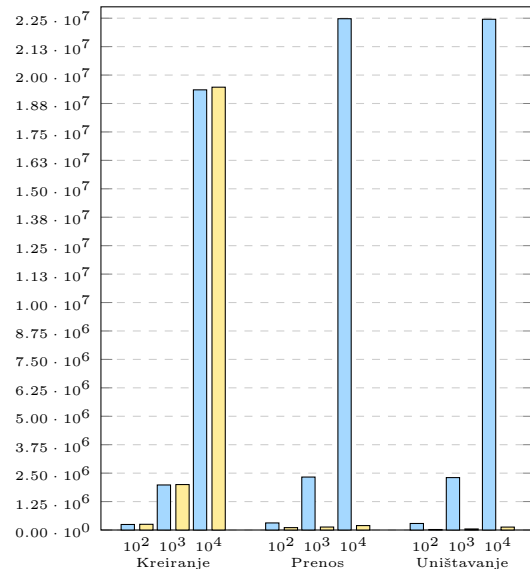


Rezultat 4: Potrošnja gasa različitih NFT standarda prilikom višestrukog uništavanja kupona u jednoj transakciji

Ponašanje operacija ERC-721A i ERC-721Psi pri velikim grupnim operacijama

Kao što je pokazano u formulama 6.4 i 6.5, prednost standarda ERC-721Psi dolazi do izražaja prilikom grupnih kreiranja. U nastavku se analizira ponašanje svake od osnovnih operacija za $n \in \{100, 1.000, 10.000\}$. Troškovi operacije kreiranja za navedene vrednosti parametra n gotovo su identični kod oba standarda, dok se kod operacija prenosa i uništavanja uočavaju značajne razlike u potrošnji gasa.

		ERC-721A	ERC-721Psi
Kreiranje	100	243.154	251.272
	1.000	1.980.198	1.998.171
	100.000	19.350.153	19.467.128
Prenos	100	311.083	105.108
	1.000	2.328.005	130.779
	100.000	22.476.835	194.654
Uništavanje	100	287.103	22.050
	1.000	2.304.024	46.000
	100.000	22.452.766	128.628



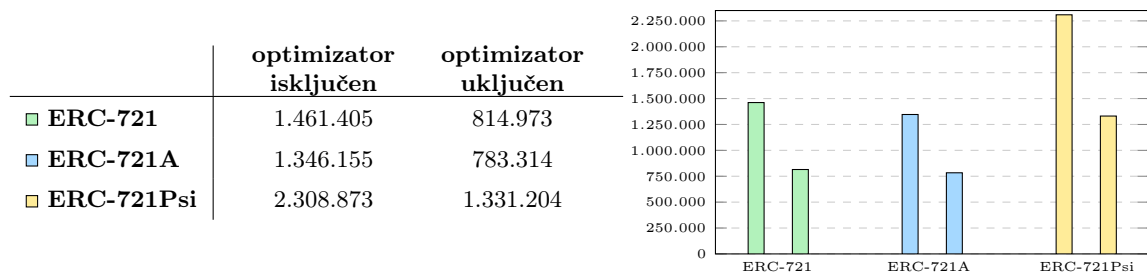
Rezultat 5: Poređenje potrošnje gasa standarda ERC-721A i ERC-721Psi prilikom velikih grupnih operacija

Postavljanje pametnih ugovora na mrežu

Prethodne analize bile su usmerene na troškove gasa koje snose korisnici, s ciljem unapređenja korisničkog iskustva, budući da upravo oni iniciraju transakcije. Sa stanovišta kompanija i preduzeća, relevantan je inicijalni trošak postavljanja pametnog ugovora na mrežu, na kojoj će korisnici naknadno izvršavati optimizovane transakcije.

Svi prethodno prikazani rezultati dobijeni su sa isključenim optimizatorom kompajlera, kako bi izveštaji o potrošnji gasa bili što precizniji i transparentniji. U nastavku se prikazuju troškovi postavljanja pametnog ugovora sa isključenim i sa uključenim optimizatorom, pri čemu je optimizator podešen na podrazumevanu vrednost od 200 pokretanja.

Kao što se može očekivati, standard ERC-721Psi ostvaruje najveći trošak postavljanja, budući da sadrži dodatnu logiku za održavanje bitmap strukture, uključujući implementaciju de Brujinovog ciklusa i prateće pretraživačke heš strukture, koje su uključene u bajtkod pri postavljanju ugovora.



Rezultat 6: Potrošnja gasa prilikom postavljanja NFT pametnog ugovora

Glava 7

Arhitektura IPFS mreže

Iako blokčejn sistemi omogućavaju pouzdano izvršavanje poslovne logike i verifikaciju stanja u distribuiranom okruženju, njihova arhitektura nije pogodna za trajno skladištenje podataka zbog ograničenja povezanih sa skalabilnošću i visokim troškovima. Iz tog razloga se u praksi izbegava direktno čuvanje velikih datoteka, kao što su slike, audio zapisi, video sadržaji i dokumenti, unutar samog blokčejna.

U savremenim distribuiranim aplikacijama blokčejn se često kombinuje sa spoljašnjim sistemima za skladištenje podataka, pri čemu se ostvaruje razdvajanje odgovornosti između sloja za verifikaciju i sloja za čuvanje sadržaja [236]. Pored toga, direktna interakcija korisnika sa blokčejn mrežom u praksi nije pogodna, pa se uvodi prezentacioni sloj koji posreduje između korisnika i pametnih ugovora, apstrahujući složenost upravljanja transakcijama i komunikacije sa mrežom.

Međutim, kako se većina savremenih web aplikacija i internet servisa oslanja na centralizovanu infrastrukturu, prezentacioni sloj i sistemi za skladištenje podataka najčešće se izvršavaju na serverskim sistemima pod upravom velikih korporativnih provajdera.

Ključna prednost centralizovanih sistema ogleda se u visokom stepenu optimizacije performansi jer se isporuka statičkog i multimedijalnog sadržaja realizuje uz minimalno kašnjenje i visok kvalitet zahvaljujući infrastrukturi zasnovanoj na lokacijskom adresiranju, kako je objašnjeno u Odeljku 3.2.2. Takav pristup omogućava veoma efikasnu distribuciju sadržaja, ali podrazumeva da operater infrastrukture zadržava potpunu kontrolu nad njegovom dostupnošću i načinom isporuke.

U takvom modelu dostupnost aplikacije i sadržaja direktno zavisi od operativnog stanja centralnih servera, pri čemu njihov prekid rada može dovesti do potpunog gubitka pristupa usluzi. Istovremeno, centralizovana arhitektura omogućava jednostavno sprovođenje kontrole i filtriranja sadržaja, čime se olakšava cenzura i ograničavanje pristupa.

Ovaj problem dodatno je izražen činjenicom da savremeni internet u velikoj meri zavisi od centralizovanih infrastrukturnih provajdera, poput mreža za isporuku sadržaja. Kompanije kao što je *Cloudflare* posreduju između korisnika i serverskih sistema, pružajući optimizaciju isporuke sadržaja, zaštitu od napada i upravljanje mrežnim saobraćajem.

Iako ovakav pristup značajno unapređuje performanse i pouzdanost servisa, istovremeno uvodi sistemski rizik, jer kvar ili prekid rada jednog takvog entiteta može uticati na dostupnost velikog broja internet servisa. Praktični incidenti pokazali su da ovakvi poremećaji mogu značajno ugroziti rad finansijskih sistema, zdravstvenih usluga i državnih servisa, što ukazuje na strukturnu ranjivost savremenog interneta [237].

Jedan od uzroka navedenih ograničenja leži upravo u činjenici da se savremeni internet do-

minantno zasniva na lokacijski orijentisanom adresiranju. Kao alternativa predlažu se sadržajno orijentisani sistemi, u kojima se podaci adresiraju na osnovu njihovog sadržaja, a ne njihove fizičke lokacije. U takvom modelu prezentacioni sloj i skladištenje sadržaja mogu biti distribuirani i kriptografski verifikovani, čime se eliminiše potreba za centralizovanim posrednicima [238].

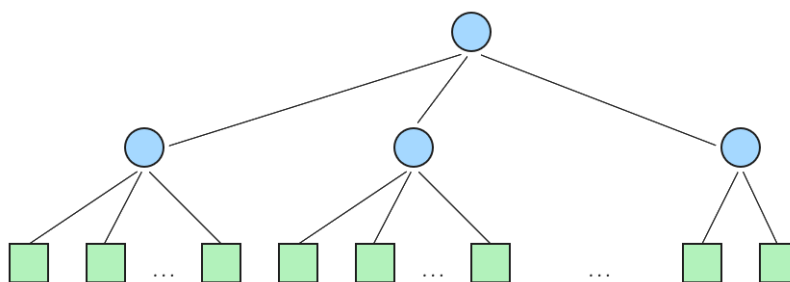
Korišćenjem decentralizovanih web aplikacija korisnici mogu direktno pristupati pametnim ugovorima i distribuiranom sadržaju, čime se ostvaruje vizija otvorenijeg i cenzuri otpornijeg interneta. Međutim, ovakav pristup često podrazumeva povećano latencijsko kašnjenje i potencijalnu degradaciju korisničkog iskustva u poređenju sa centralizovanim rešenjima, kao posledicu distribuirane prirode skladištenja, isporuke sadržaja i izvršavanja aplikacija.

Međuplanetarni fajl sistem (eng. *InterPlanetary File System, IPFS*) [239] predstavlja distribuirani sistem zasnovan na mreži ravnopravnih čvorova, namenjen skladištenju i razmeni podataka bez centralne tačke upravljanja. U kontekstu sistema e-trgovine zasnovanih na blokčejnu, upotreba IPFS-a nije neophodna, ali predstavlja značajno unapređenje u pogledu distribuiranosti i efikasnosti skladištenja podataka, čime se omogućava izgradnja potpuno decentralizovanih platformi.

Podrazumevani način objavljivanja i pribavljanja sadržaja u IPFS mreži podrazumeva da korisnik lokalno pokrene IPFS servis u pozadini (eng. *daemon*), koji funkcioniše kao **koordinacioni** čvor u procesu razmene podataka. Ovak čvor uspostavlja izlazne (eng. *outbound*) konekcije ka drugim čvorovima u mreži, dok istovremeno može prihvatati dolazne (eng. *inbound*) konekcije. Alternativno, korisnik može koristiti spoljne servise koji omogućavaju pristup IPFS mreži bez pokretanja lokalnog čvora.

7.1 Strukturiranje i adresiranje sadržaja

Prilikom postavljanja fajla na IPFS mrežu, sadržaj fajla se deli na manje blokove veličine 256KB-1MB. Dobijeni blokovi se zatim organizuju u strukturu podataka tipa *Merkle Directed Acyclic Graph (Merkle DAG)* [240], gde listovi grafa predstavljaju sirove blokove podataka koji sadrže isključivo segmente originalnog binarnog sadržaja. Koren grafa predstavlja čvor koji enkapsulira metapodatke o fajlu (npr. tip objekta i ukupnu veličinu), zajedno sa referencama ka podređenim čvorovima. Ukoliko broj blokova prevazilazi kapacitet korenog čvora, referenciranje se hijerarhijski organizuje kroz dodatne međučvorove, čime se formira balansirani Merkle DAG.

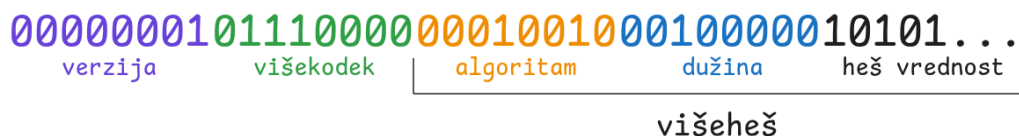


Slika 7.1: Formiranje Merkle DAG strukture iz sadržaja fajla.

Jedinstveni identifikator sadržaja (eng. *Content Identifier, CID*) je nepromenljiv i formira se kombinovanjem sledećih informacija:

1. Verzija identifikatora, radi obezbeđivanja kompatibilnosti unapred.
2. Višekodek (eng. *multicodec*), koji označava format i tip kodiranja sadržaja.

3. Višeheš (eng. *multihash*), koji sadrži informaciju o korišćenoj heš funkciji, dužini heš vrednosti i samoj heš vrednosti korenog čvora grafa.
4. Višebazno (eng. *multibase*) enkodiranje, koje se primenjuje samo prilikom prikaza CID-a u obliku tekstualne niske.



Slika 7.2: Komponente CID identifikatora.

Važno je napomenuti da identični fajlovi mogu imati različite CID identifikatore ukoliko su generisani korišćenjem različitih parametara kodiranja, kao što su drugačija verzija identifikatora, drugačiji tip formata ili drugačija heš funkcija, a ako se koriste isti parametri, tada isti sadržaj uvek rezultuje istim identifikatorom. Ukoliko više korisnika u mreži dodaje isti fajl, generisaće se isti CID, što omogućava ponovnu upotrebu već postojećih blokova podataka, tj. implicitnu deduplikaciju sadržaja na nivou blokova.

Dodatno, sadržajno adresiranje omogućava kriptografsku proveru integriteta podataka. Pošto CID uključuje heš vrednost korenog čvora strukture podataka, svaka promena u sadržaju rezultuje drugačijom heš vrednošću, a samim tim i drugačijim identifikatorom. Na taj način korisnik može lokalno proveriti da li preuzeti sadržaj odgovara očekivanom identifikatoru.

Međutim, u mnogim praktičnim scenarijima javlja se potreba za ažuriranjem sadržaja, na primer u slučaju web stranica postavljenih na IPFS-u. Budući da je CID nepromenljiv identifikator koji direktno zavisi od sadržaja podataka, svaka izmena fajla rezultuje generisanjem novog CID-a. Zbog toga održavanje referenci na najnoviju verziju sadržaja može postati nepraktično.

Radi rešavanja ovog problema koristi se međuplanetarni sistem imena (eng. *InterPlanetary Name System, IPNS*), koji predstavlja distribuirani sistem imenovanja zasnovan na kriptografskim ključevima. U okviru ovog sistema identitet prostora *pokazivača* definiše se parom javnog i privatnog ključa, pri čemu javni ključ određuje jedinstveni prostor imena u kojem je moguće ažurirati promenljive reference ka nepromenljivim CID identifikatorima.

Ažuriranje fajla u okviru IPNS zapisa zapravo predstavlja ažuriranje reference datog imena tako da pokazuje na željeni CID, pri čemu se autentičnost ažuriranja obezbeđuje privatnim ključem odgovarajućeg identiteta. Na taj način IPNS omogućava postojanje stabilnih i promenljivih identifikatora koji mogu referencirati različite verzije sadržaja tokom vremena.

7.2 Distribuirano pribavljanje sadržaja

Na osnovu CID-a moguće je deterministički rekonstruisati strukturu grafa i time pribaviti originalni fajl, pod uslovom da su svi njegovi blokovi dostupni u mreži. Na ovaj način IPFS zamenjuje lokacijski zasnovano adresiranje sadržajnim adresiranjem (eng. *content-based addressing*). Korisnik ne zahteva podatak sa određene fizičke ili mrežne lokacije, već specificira kriptografski identifikator željenog sadržaja. Time se konceptualni fokus pomera sa pitanja „gde se podatak nalazi” na pitanje „koji podatak je potreban”, čime se povećava otpornost sistema na cenzuru i prekide rada pojedinačnih čvorova.

Razmena blokova između povezanih čvorova realizuje se korišćenjem protokola *Bitswap*, koji predstavlja mehanizam za distribuiranu razmenu blokova između ravnopravnih čvorova. Prilikom zahteva za određenim sadržajem lokalni čvor formira listu željenih blokova (eng. *want-list*), koja sadrži identifikatore potrebnih blokova. Ova lista se **propagira** putem poruka sa svim trenutno povezanim čvorovima, dok povezani čvorovi razmenjuju sopstvene liste želja.

Proces pribavljanja sadržaja započinje dodavanjem CID identifikatora korenog čvora traženog sadržaja u listu željenih blokova. Nakon pribavljanja ovog bloka iz njegove strukture izdvajaju se reference na podređene blokove, koji se zatim dodaju u listu željenih blokova. Na taj način se postupno otkriva kompletna struktura grafa i proširuje skup blokova koje je potrebno pribaviti.

Preuzimanje blokova realizuje se paralelno sa više čvorova ukoliko je sadržaj dostupan na više lokacija u mreži, čime se povećava stepen paralelizacije i efikasnost prenosa podataka.

Ukoliko traženi blokovi nisu dostupni među trenutno povezanim čvorovima, IPFS koristi distribuiranu heš tabelu zasnovanu na protokolu *Kademlia DHT* [241] radi pronalaženja čvorova koji mogu obezbediti traženi sadržaj. Pretraga u ovoj strukturi zasniva se na iterativnom pronalaženju čvorova čiji identifikatori imaju manju udaljenost u odnosu na traženi ključ. Ovakav pristup omogućava pronalaženje odgovarajućih čvorova uz logaritamsku složenost u odnosu na veličinu mreže.

Nakon pronalaženja čvorova koji mogu obezbediti tražene blokove, lokalni čvor uspostavlja izlaznu konekciju ka njima, nakon čega se razmena blokova nastavlja korišćenjem protokola *Bitswap*. Proces se ponavlja sve dok svi potrebni blokovi ne budu pribavljeni, čime se omogućava rekonstrukcija originalnog fajla.

Blokovi podataka koji se preuzmu tokom procesa pribavljanja sadržaja privremeno se skladište na lokalnom čvoru, čime taj čvor može postati dodatni izvor podataka za druge učesnike u mreži. Na taj način IPFS mreža prirodno ostvaruje efekat distribuirane replikacije sadržaja kroz proces razmene blokova između ravnopravnih čvorova. Međutim, blokovi se smatraju privremenim i mogu biti uklonjeni tokom procesa prikupljanja otpatka (eng. *garbage collection*). Ovaj proces periodično oslobađa prostor u lokalnom skladištu uklanjanjem blokova koji nisu dugo referencirani.

Ukoliko korisnik ne želi da pokreće lokalni IPFS čvor, pristup sadržaju moguće je ostvariti putem IPFS pristupnika (eng. *gateway*). Pristupnik predstavlja servis koji izlaže standardizovani HTTP interfejs za pribavljanje sadržaja adresiranog putem CID identifikatora. Na ovaj način IPFS sadržaj postaje dostupan kroz konvencionalni web protokol, što omogućava pristup podacima i korisnicima koji ne učestvuju direktno u mreži ravnopravnih korisnika.

Pristupnik prima HTTP zahtev koji sadrži CID identifikator u URL adresi, nakon čega pokušava da pronađe odgovarajući sadržaj. Ukoliko pristupnik već poseduje tražene blokove u lokalnom skladištu, sadržaj se odmah vraća klijentu. U suprotnom, ponašanje zavisi od konfiguracije pristupnika:

1. Rekurzivni pristupnik pokušava da pribavi traženi sadržaj iz IPFS mreže pre nego što ga prosledi korisniku.
2. Nerekurzivni pristupnik vraća poruku o nedostupnosti podataka.

Oslanjanje na pristupnike trećih strana uvodi određene kompromise u pogledu decentralizacije sistema. Budući da pristupnike obično održava jedan administrativni entitet, oni mogu predstavljati tačke centralizovane kontrole nad pristupom sadržaju. Operateri pristupnika imaju tehničku mogućnost da evidentiraju zahteve korisnika i sadržaj koji se pribavlja, što može imati implikacije na privatnost. Pored toga, veliki broj korisnika može koristiti isti javni pristupnik, što potencijalno dovodi do zagušenja i smanjenja propusnosti.

Kao potencijalno rešenje razvijaju se tzv. pristupnici bez poverenja (eng. *trustless gateways*), koji omogućavaju verifikaciju sadržaja direktno na strani klijenta. U tom slučaju pregledač može kriptografski proveriti da li preuzeti sadržaj odgovara očekivanom CID identifikatoru, čime se smanjuje potreba za poverenjem u operatera pristupnika. Ipak, podrška za ovakav način rada još uvek nije široko implementirana u savremenim web pregledačima.

7.3 Dugoročno skladištenje podataka

Replikacija sadržaja objavljenog u IPFS mreži nije automatski zagarantovana. Podaci ostaju dostupni u mreži sve dok ih najmanje jedan čvor poseduje i aktivno deli sa ostalim učesnicima, pri čemu je u najjednostavnijem slučaju jedini takav čvor onaj koji je sadržaj prvobitno objavio.

Ovakav model čuvanja podataka predstavlja jedno od osnovnih ograničenja IPFS mreže, jer sam protokol omogućava efikasno adresiranje i distribuciju sadržaja, ali ne rešava problem njegove dugoročne dostupnosti. Drugim rečima, IPFS ne uvodi mehanizme kojima bi učesnici mreže bili ekonomski podstaknuti da trajno skladište podatke drugih korisnika.

Ukoliko korisnik ne želi da održava sopstveni IPFS čvor stalno aktivnim, dugoročna dostupnost sadržaja može se povećati korišćenjem usluge kačenja (eng. *pinning service*), koja zadržava određene blokove podataka na infrastrukturi treće strane kako oni ne bi bili uklonjeni iz lokalnog skladišta. U praksi se ovakve usluge najčešće komercijalno naplaćuju.

Međutim, korišćenje usluge kačenja u suštini predstavlja oblik smeštanja podataka (eng. *hosting*) IPFS sadržaja na infrastrukturi treće strane. Dugoročna dostupnost podataka zavisi od toga da će pružalac usluge nastaviti da čuva sadržaj, pri čemu korisnik nema proverljiv mehanizam kojim bi mogao potvrditi da se podaci zaista i dalje skladište [242].

Radi prevazilaženja ovog ograničenja razvijeni su decentralizovani blokčejn sistemi za skladištenje podataka zasnovani na *kriptoekonomskim mehanizmima*, koji su kompatibilni sa IPFS ekosistemom. Najpoznatiji predstavnik ove grupe sistema jeste *Filecoin* (Fajlkoin) [243], čija je osnovna karakteristika uspostavljanje *decentralizovanog tržišta skladišnog prostora* na kome učesnici mreže ostvaruju finansijske podsticaje za pouzdano dugoročno čuvanje podataka.

Ovo tržište zasniva se na **potražnji** korisnika kojima je potreban skladišni prostor i **ponudi** dobavljača skladišta (eng. *storage providers*), koji preuzimaju obavezu da podatke čuvaju tokom ugovorenog vremenskog perioda. Za pružanje ove usluge dobavljači skladišta ostvaruju naknadu u primarnoj kriptovaluti mreže, dok blokčejn automatski sprovodi pravila protokola putem kriptografske verifikacije i odgovarajućih ekonomskih podsticaja.

Prilikom sklapanja ugovora o skladištenju dobavljač skladišta generiše kriptografski dokaz da je za primljene podatke napravio jedinstvenu fizičku repliku (eng. *Proof of Replication*) [244]. Nakon toga protokol u pravilnim vremenskim intervalima generiše nasumične izazove (eng. *challenges*), na koje dobavljač skladišta mora odgovoriti dostavljanjem validnog kriptografskog dokaza da se podaci i dalje nalaze u njegovom lokalnom skladištu (eng. *Proof of Spacetime*) [245]. Ukoliko nije u mogućnosti da proizvede odgovarajući dokaz, smatra se da nije ispunio svoje obaveze, nakon čega protokol automatski primenjuje finansijske kazne definisane pravilima mreže.

Za razliku od usluge kačenja, dobavljači skladišta u Fajlkoin mreži nisu prvenstveno namenjeni brzom isporučivanju sadržaja, već njegovom dugoročnom čuvanju. Ukoliko traženi sadržaj nije dostupan na nekom IPFS čvoru, može se poslati zahtev za preuzimanje (eng. *retrieval request*) ka Fajlkoin mreži, pri čemu odgovarajući dobavljač skladišta isporučuje podatke nazad u IPFS ekosistem ili direktno korisniku.

7.4 Korisnički pristup decentralizovanim aplikacijama

Iako IPFS omogućava decentralizovano skladištenje i distribuciju sadržaja, potpuno decentralizovana aplikacija zahteva i mehanizam putem kojeg korisnici mogu bezbedno upravljati svojim identitetom i ostvarivati interakciju sa pametnim ugovorima. U tom kontekstu posebnu ulogu imaju kriptografski novčanici, koji omogućavaju čuvanje privatnih ključeva i autorizaciju transakcija bez potrebe za centralizovanim posrednikom.

Da bi platforma zadržala svoj decentralizovani karakter, neophodno je da korisnici zadrže kontrolu nad sopstvenim privatnim ključevima. Ovakav pristup odgovara principu potpune decentralizacije sistema (eng. *end-to-end decentralization*), pri kome se poverenje ne prenosi na centralizovane posrednike, već ostaje u rukama krajnjih korisnika.

Time se istovremeno ostvaruje i princip korisničkog suvereniteta nad kriptografskim identitetom (eng. *self-custody*). Ukoliko platforma upravlja privatnim ključevima korisnika (eng. *custodial model*), narušava se princip minimizacije poverenja (eng. *trust minimization*), jer korisnici postaju zavisni od centralizovanog entiteta [246].

U cilju obezbeđivanja decentralizovanog pristupa pametnim ugovorima, platforme koriste različite mehanizme potpisivanja transakcija (eng. *signer*), koji mogu biti realizovani u sledećim oblicima [247]:

1. Topli novčanik (eng. *hot wallet*) u obliku ekstenzije pretraživača koja skladišti privatne ključeve u okviru korisničkog okruženja (npr. *MetaMask*, *Phantom*), pri čemu se potpisivanje transakcija vrši direktno iz pretraživača.
2. Hladni novčanik (eng. *cold wallet*), odnosno hardverski uređaj koji čuva privatne ključeve izolovano od mreže (npr. *Ledger*, *Trezor*), gde se transakcija potpisuje na samom uređaju, a zatim se potpisana transakcija prosleđuje mreži putem posredničkog kanala (npr. USB ili QR kod).
3. Hibridni novčanik (eng. *hybrid wallet*) koji kombinuje karakteristike toplih i hladnih novčanika, gde se deo funkcionalnosti, poput interakcije sa aplikacijom i pripreme transakcije, izvršava u softverskom okruženju, dok se kritične operacije potpisivanja obavljaju u sigurnom, izolovanom modulu, čime se postiže kompromis između upotrebljivosti i bezbednosti.

Hardverski novčanici se mogu posmatrati kao najsigurnija varijanta, budući da privatni ključevi nikada ne napuštaju namenski uređaj. Na taj način smanjuje se rizik od kompromitacije usled prisustva zlonamernog softvera na korisničkom računaru, koji bi u suprotnom mogao omogućiti neovlašćeno potpisivanje transakcija ili pristup privatnim ključevima.

Iako IPFS i blokčejn sistemi koriste kriptografske identifikatore radi obezbeđivanja integriteta i jedinstvenosti resursa, takvi identifikatori nisu pogodni za svakodnevnu upotrebu od strane krajnjih korisnika. Adrese novčanika, CID identifikatori i drugi kriptografski identifikatori predstavljeni su dugim heksadecimalnim nizovima karaktera koji su podložni greškama prilikom unosa, a i nije ih lako zapamtiti.

Iz tog razloga razvijeni su sistemi imenovanja, među kojima se izdvaja usluga imenovanja Eterijuma (eng. *Ethereum naming service*, *ENS*) [248] kao decentralizovani sistem koji mapira ljudima čitljiva imena na blokčejn adrese ili lokacije sadržaja.

ENS se oslanja na dva osnovna tipa pametnih ugovora za upravljanje domenima:

1. Ugovor o registru, koji održava evidenciju svih domena i definiše pravila njihovog upravljanja, uključujući vlasništvo i delegiranje poddomena.
2. Ugovor o razrešavanju, koji omogućava mapiranje domena na odgovarajuće resurse, kao što su adrese novčanika ili lokacije decentralizovanog sadržaja.

Proces razrešavanja ENS domena podrazumeva slanje zahteva ugovoru o registru, koji vraća adresu odgovarajućeg ugovora o razrešavanju, nakon čega se putem tog ugovora dobija konkretan resurs povezan sa datim domenom.

Kombinacijom ENS sistema, decentralizovanog skladištenja sadržaja i pametnih ugovora moguće je uspostaviti model u kome se kontrola nad digitalnim resursima i funkcionalnostima platforme prenosi direktno putem blokčejn transakcija. Time se značajno smanjuje potreba za tradicionalnim administrativnim procedurama zasnovanim na centralizovanim registrima i posrednicima. Ipak, praktična primena ovakvog pristupa i dalje zavisi od regulatornog okvira i pravnog priznavanja blokčejn transakcija u različitim pravnim sistemima, što predstavlja jedno od značajnih otvorenih pitanja i pravaca budućih istraživanja [249].

Glava 8

Diskusija

U prethodnim poglavljima predstavljen je koncept dizajna sistema elektronske trgovine zasnovanog na proizvodima i uslugama sa jedinstvenim identifikatorima, realizovan primenom NFT standarda na Eterijum blokčejnu, koji predstavlja najzastupljeniju platformu za razvoj decentralizovanih aplikacija. Dodatno, analizirane su mogućnosti primene sadržajno adresiranih sistema za očuvanje suvereniteta nad kriptografskim identitetom korisnika.

Iako predložena arhitektura omogućava eliminisanje pojedinačnih tačaka otkaza i povećava autonomiju korisnika u odnosu na centralizovane platforme, njena praktična primena istovremeno otvara brojna tehnička, regulatorna i bezbednosna pitanja koja prevazilaze obim implementacionog dela ovog rada.

Bez obzira na tehničke prednosti decentralizovanih sistema, njihova šira primena u oblasti elektronske trgovine u velikoj meri zavisi od regulatornog okruženja. Postojeći pravni okviri uglavnom su razvijani za centralizovane informacione sisteme u kojima postoji jasno definisan upravljač podataka i odgovorna pravna strana.

Nasuprot tome, decentralizovane mreže raspodeljuju odgovornost raspodeljuju između velikog broja učesnika mreže, što otvara pitanja usklađenosti sa propisima koji se odnose na zaštitu podataka o ličnosti, pravo na brisanje podataka, odgovornost za nezakonit sadržaj, zaštitu potrošača i prekogranični prenos podataka [250].

Zbog toga regulatorna neizvesnost predstavlja jedan od ključnih faktora koji će određivati stepen prihvatanja decentralizovanih tehnologija u komercijalnim informacionim sistemima. Pored pravnih izazova, dodatni razlog za oprez predstavlja mogućnost zloupotrebe decentralizovanih infrastruktura za distribuciju nezakonitog sadržaja, prikrivanje identiteta učesnika ili druge oblike kriminala, zbog čega regulatorni okvir ima značajnu ulogu u njihovoj široj primeni [251].

U ovom poglavlju razmatraju se mogući pravci daljeg razvoja predložene arhitekture, uključujući proširenje funkcionalnosti NFT standarda, integraciju blokčejna sa spoljnim izvorima podataka, kao i specifičnosti primene decentralizovanog prezentacionog sloja zasnovanog na IPFS mreži.

8.1 Proširene funkcionalnosti NFT standarda u sistemima e-trgovine

NFT standardi takođe predviđaju mehanizam tzv. zakačaljki (eng. *hooks*), odnosno funkcija koje se pozivaju pre i nakon svake osnovne operacije, kroz interne funkcije `_beforeTokenTransfer` i

_afterTokenTransfer. Ove funkcije su podrazumevano prazne, ali omogućavaju proširenje osnovnog ponašanja bez izmene standardne logike. Na primer, pre izvršenja prenosa može se implementirati dodatna provera autorizacije, kojom se potvrđuje da pošiljalac ili primalac pripada unapred definisanoj grupi korisnika.

Pored osnovnih operacija kreiranja, prenosa i uništavanja, NFT standardi predviđaju i funkciju odobravanja (eng. *approve*), koja omogućava vlasniku NFT-a da ovlasti drugu adresu da prenese konkretan token u njegovo ime. Time se uvodi mehanizam imenovanja prava raspolaganja, bez prenosa vlasništva. Zbog prilagodljive prirode pametnih ugovora, ova funkcionalnost može se proširiti ili modifikovati, na primer tako da ovlašćena adresa može izvršiti i operaciju uništavanja tokena u ime vlasnika, u skladu sa poslovnom logikom platforme.

NFT aplikacije se oslanjaju na mehanizam metapodataka, pri čemu se dodatne informacije o tokenu čuvaju na eksternim servisima u JSON formatu, pri čemu ti podaci nisu direktno dostupni unutar same blokčejn mreže. Pristup tim podacima za određeni identifikator ostvaruje se putem funkcije *tokenURI(uint256 tokenId)*, koja vraća referencu (najčešće URL) ka resursu dostupnom van mreže [252]. U zavisnosti od arhitekture sistema, servisi za skladištenje mogu biti centralizovani (npr. baze podataka) ili decentralizovani (npr. IPFS).

Budući da se transakcione naknade plaćaju u primarnoj kriptovaluti blokčejn mreže, korisnik se može naći u situaciji da poseduje NFT, ali nema dovoljno sredstava za pokrivanje troškova izvršavanja transakcije. Ovaj problem moguće je ublažiti fleksibilnijim modelom interakcije primenom ekstenzije za *dozvolu* [253] (eng. *permit*), koja je još u fazi standardizacije. Navedeni mehanizam omogućava vlasniku tokena da van mreže potpiše kriptografsku poruku kojom odobrava određenu operaciju, prosledi je drugoj strani, koja na osnovu tog potpisa može izvršiti odobrenu operaciju u njegovo ime i snositi troškove transakcione naknade.

Dodatno, ukoliko je NFT tržišno prenosiv, moguće je u sam pametni ugovor ugraditi mehanizam kojim se pri svakoj sekundarnoj prodaji ili pretprodaji automatski alokira određeni procenat sredstava ka unapred definisanoj adresi [254]. Ovakav pristup omogućava implementaciju modela trajnih autorskih naknada (eng. *royalties*), čime se obezbeđuje kontinuirano finansiranje izdavaoca proizvoda. Time se finansijska logika distribucije prihoda ugrađuje direktno u izvršni kod pametnog ugovora, čime se eliminiše potreba za posrednicima i povećava transparentnost obračuna.

Za razliku od prethodno opisanih proširenja koja su specifično vezana za NFT standard, u praksi se široko primenjuju i generičke ekstenzije pametnih ugovora koje unapređuju bezbednost, kontrolu pristupa i upravljanje životnim ciklusom sistema. Ove komponente nisu ograničene na NFT logiku, već predstavljaju opšte obrasce modularnog dizajna primenljive na različite tipove pametnih ugovora. Biblioteka *OpenZeppelin* obezbeđuje skup standardizovanih, bezbednosno proverenih i optimizovanih implementacija takvih ekstenzija, koje su postale industrijski standard.

Najčešće korišćene ekstenzije ove vrste uključuju:

1. *Pausable* - mehanizam koji omogućava privremenu obustavu izvršavanja odabranih funkcija ugovora u vanrednim okolnostima, bez trajne izmene stanja sistema.
2. *AccessControl* - model kontrole pristupa zasnovan na ulogama, kojim se precizno definišu privilegije i nadležnosti različitih uloga unutar sistema.
3. *ReentrancyGuard* - zaštitni mehanizam koji sprečava ponovni ulazak u funkciju pre završetka njenog prethodnog izvršavanja, čime se sprečavaju napadi zasnovani na rekurzivnim pozivima [255].

4. *MultiSigner* - obrazac kolektivnog odlučivanja u kojem je za izvršavanje osetljivih operacija potrebno više nezavisnih ovlašćenih potpisa tzv. višepotpisnih novčanika, čime se smanjuje rizik centralizovanog upravljanja.

Pored toga, moguće je delimično ili u određenim slučajevima u potpunosti ograničiti transparentnost javnih blokčejn mreža primenom potpuno homomornog šifrovanja (eng. *fully homomorphic encryption*), koje omogućava izvršavanje operacija direktno nad šifrovanim podacima bez potrebe za njihovim dešifrovanjem [256]. U ovom modelu pametni ugovori primaju šifrovane ulaze, izvršavaju operacije nad njima i održavaju šifrovano stanje, bez pristupa originalnim vrednostima, koje mogu biti selektivno dešifrovane od strane ovlašćenih aktera. Tokom obrade transakcija u okviru konsenzusnog mehanizma operacije se izvršavaju nad šifrovanim vrednostima, pri čemu svi čvorovi deterministički dolaze do istog rezultata. Ovakav pristup predstavlja značajan korak ka širem usvajanju blokčejn tehnologije, iako se i dalje nalazi u fazi aktivnog razvoja.

Korišćenjem ovih mehanizama pametnih ugovora, distribuirana platforma za e-trgovinu može obezbediti širok spektar funkcionalnosti, uz zadržavanje formalne konzistentnosti i sigurnosnih garancija javne blokčejn mreže gde se poslovna pravila mogu implementirati direktno na nivou ugovora.

8.2 Obrasci modularnog dizajna pametnih ugovora

U arhitekturi platforme moguće je predvideti postojanje više međusobno povezanih pametnih ugovora, pri čemu pojedini ugovori preuzimaju specifične sistemske uloge. Ugovor o registru [257] (eng. *registry contract*) predstavlja pametni ugovor koji funkcioniše kao decentralizovana baza podataka, čuvajući mape između imena, identifikatora ili drugih podataka i odgovarajućih adresa pametnih ugovora. Na taj način omogućava centralizovano usmeravanje poziva ka postojećim NFT kolekcijama.

Ukoliko je potrebno masovno kreiranje novih instanci kojima upravlja ugovor o registru, mogu se primeniti sledeći obrasci:

1. **Fabrički ugovor** (eng. *factory contract*) je pametni ugovor čija je osnovna funkcija postavljanje novih ugovora na mrežu. Njegova prednost ogleda se u mogućnosti dinamičkog kreiranja proizvoljnih instanci po potrebi, pri čemu je za postavljanje dovoljan bajtkod ugovora koji se implementira.
2. **Minimalni zastupnički ugovor** [258] (eng. *minimal proxy contract*) predstavlja specifičan slučaj kada se kreiraju identične instance ugovora, poznate kao klonovi (eng. *clones*). U ovom obrascu izvršna logika nalazi se u jedinstvenom implementacionom ugovoru (eng. *implementation contract*), dok klonovi poseduju sopstveno skladište podataka. Izvršni kod se delegira ka implementacionom ugovoru, čija se adresa nalazi u bajtkodu klona. Budući da je izvršni deo bajtkoda minimalnog zastupničkog ugovora dugačak svega 45 bajtova, troškovi postavljanja značajno su niži u odnosu na potpuno postavljanje ugovora sa identičnom logikom.

Iako je kod pametnih ugovora po svojoj prirodi neizmenljiv nakon njihovog postavljanja na mrežu, ovo ograničenje može se prevazići primenom nadogradivih arhitektura [259]. Uobičajeni pristup podrazumeva razdvajanje skladišne i izvršne logike kroz dva međusobno povezana ugovora:

1. **Zastupnički ugovor** [260], koji čuva skladišne promenljive i predstavlja stabilnu tačku interakcije sa korisnicima.

2. **Implementacioni ugovor**, koji sadrži izvršni kod i može se zameniti u procesu nadogradnje bez promene adrese zastupničkog ugovora.

U zavisnosti od načina implementacije nadogradnje razlikuju se transparentni zastupnički ugovor (eng. *Transparent Proxy*) i univerzalni nadogradivi zastupnički ugovor (eng. *Universal Upgradeable Proxy Standard, UUPS*).

Kod kompleksnijih sistema mogu se primeniti dodatni obrasci:

1. U slučaju postojanja većeg broja zastupničkih ugovora, radi pojednostavljenja nadogradnje može se primeniti obrazac svetionika (eng. *beacon*). U tom slučaju zastupnički ugovori u svom bajtkodu čuvaju adresu ugovora svetionika, koji centralno upravlja adresom aktuelne implementacije. Izmena implementacije na nivou ugovora svetionika automatski utiče na sve povezane zastupničke ugovore.
2. Kada je potrebno modularizovati implementacioni ugovor i prevazići ograničenje maksimalne veličine bajtkoda, može se primeniti dijamantski obrazac [261] (eng. *diamond pattern*). Ovaj pristup omogućava razdvajanje funkcionalnosti na više međusobno povezanih komponenti (eng. *facets*), pri čemu centralni ugovor (eng. *diamond*) delegira pozive odgovarajućim modulima. Time se omogućava nadogradnja pojedinačnih delova sistema i fleksibilno proširenje funkcionalnosti.

U praksi, proces nadogradnje i upravljanja implementacijama često se organizuje kroz višepotpisnih novčanika ili kroz DAO mehanizme, kako bi se smanjila centralizovana kontrola i povećala decentralizacija odluka o promenama koda, posebno u slučajevima kada promena utiče na velike korisničke baze.

Ovakav arhitektonski pristup omogućava da pametni ugovori preuzmu ulogu aplikativnog sloja platforme, pri čemu korisnici, nakon autentikacije putem kriptografskih ključeva, neposredno komuniciraju sa blokčejn infrastrukturom, dok se logika sistema izvršava u determinističkom okruženju distribuirane mreže.

8.3 Integracija blokčejn sistema sa spoljašnjim izvorima podataka

Blokčejn sistem predstavlja zatvoreno determinističko okruženje u kome svaki čvor, nakon prijema novog bloka, mora izračunati identično globalno stanje sistema. Ovakav model obezbeđuje visok stepen integriteta i neizmenljivosti podataka, jer naknadna manipulacija već potvrđenim zapisima praktično nije moguća. Međutim, posledica takve arhitekture jeste *odsustvo ugrađenih mehanizama za pristup informacijama van same mreže*, što značajno ograničava funkcionalni domet pametnih ugovora.

Da bi se podaci javno dostupni na klasičnom internetu uveli u blokčejn sistem i učinili dostupnim pametnim ugovorima, koristi se mehanizam tzv. proročišta [262] (eng. *oracles*), koji predstavlja posrednički sloj između blokčejn sistema i spoljašnjih izvora podataka. S obzirom na to da blokčejn ne može direktno pozivati eksterne API servise, proročište tu operaciju izvršava van mreže, preuzima relevantne podatke iz eksternih izvora, obrađuje ih i potom putem transakcije upisuje rezultat u sopstveni pametni ugovor na blokčejn mreži. Ostali pametni ugovori zatim pristupaju tim podacima pozivanjem funkcija ugovora proročišta.

Osnovni izazov ovakvog modela jeste obezbeđivanje poverenja i očuvanje decentralizacije. Ukoliko decentralizovani blokčejn sistem zavisi od centralizovanog proročišta, uvodi se jedinstvena tačka otkaza, što je u literaturi poznato kao *problem proročišta* (eng. *The Oracle Problem*). Kompromitovano proročište ili nepouzdan izvor podataka može uticati na izvršavanje svih pametnih ugovora koji se na njega oslanjaju, što potencijalno dovodi do bezbednosnih incidenata i finansijskih gubitaka [263].

Radi prevazilaženja ovog problema razvijene su decentralizovane mreže proročišta [264] (eng. *Decentralized Oracle Networks, DON*), koje agregiraju podatke iz više nezavisnih čvorova i različitih izvora. Agregacijom i mehanizmima konsenzusa umanjuje se rizik manipulacije pojedinačnog entiteta, čime se povećavaju pouzdanost i bezbednost sistema.

Pored dostavljanja eksternih podataka, decentralizovane mreže proročišta omogućavaju i druge funkcionalnosti, uključujući generisanje verifikabilne slučajnosti, automatizaciju izvršavanja pametnih ugovora, međulančanu komunikaciju, izvršavanje određenih proračuna van lanca, kao i mehanizme dokaza o rezervama. Time se značajno proširuje primenljivost blokčejn sistema u realnim poslovnim okruženjima.

Mehanizmi proročišta su danas visoko razvijeni kada je reč o javno dostupnim podacima. Međutim, u savremenim primenama javlja se potreba za pouzdanim uvozom privatnih korisničkih podataka iz postojećih informacionih sistema, kao što su bankarski servisi, sistemi elektronskog identiteta, društvene mreže, servisi za elektronsku trgovinu i drugi centralizovani internet servisi.

U okviru standardnog TLS protokola, nakon uspostavljanja sigurne komunikacione sesije, podaci koje klijent preuzima od servera jesu kriptografski zaštićeni tokom prenosa, ali nisu kriptografski dokazivi trećoj strani. Drugim rečima, klijent ne poseduje dokaz koji bi nezavisno potvrdio da su konkretni podaci zaista potekli od određenog servera.

Nadogradnja TLS protokola, poznata kao zkTLS [265], uvodi mehanizam generisanja kriptografskog dokaza kojim se potvrđuje da su pribavljeni podaci zaista dostavljeni od strane servera koji poseduje važeći digitalni sertifikat. Ovako definisan pristup omogućava da se selektivno izdvojene informacije iz postojećih internet servisa, koji već koriste TLS infrastrukturu, pouzdano prenesu u blokčejn sistem zajedno sa kriptografskim dokazom o njihovom poreklu, bez potrebe za modifikacijom samih servisa. Na taj način ostvaruje se visok potencijal za globalno skaliranje blokčejn sistema.

Ipak, bezbednosni model ostaje uslovljen pretpostavkom ispravnosti i integriteta eksternih servisa koji generišu podatke, kao i poverenjem u autoritete za sertifikate u okviru TLS infrastrukture. Stoga se sigurnost sistema delimično oslanja na spoljne entitete, što predstavlja dodatnu tačku rizika u ukupnoj arhitekturi.

8.4 Decentralizovano skladištenje podataka

Za razliku od tradicionalnih web aplikacija, kod kojih se sadržaj preuzima sa unapred poznatog servera, IPFS koristi distribuiranu mrežu čvorova u kojoj se podaci pronalaze na osnovu jedinstvenog identifikatora sadržaja (CID). Takav pristup doprinosi otpornosti sistema i eliminiše zavisnost od centralnog servera, ali istovremeno uvodi određene izazove koji mogu uticati na korisničko iskustvo i praktičnu primenu.

Jedan od najvažnijih izazova odnosi se na vreme odaziva prilikom pristupa sadržaju. Za razliku od centralizovanih mreža za isporuku sadržaja (CDN), kod kojih se sadržaj najčešće isporučuje sa geografski najbližeg servera, IPFS prvo mora pronaći odgovarajuće čvorove, nakon čega započinje

samo preuzimanje podataka. Vreme potrebno za preuzimanje datoteke zavisi od broja čvorova koji poseduju traženi sadržaj, njihove dostupnosti, kao i mrežne udaljenosti između korisnika i odgovarajućih čvorova. Ukoliko se određeni sadržaj nalazi na malom broju čvorova, vreme odziva može biti značajno veće.

Iako ovakvo ponašanje ne utiče na funkcionalnu ispravnost sistema, u elektronskoj trgovini čak i relativno mala povećanja vremena učitavanja mogu negativno uticati na korisničko iskustvo, imajući u vidu da korisnici očekuju brz odziv nezavisno od tehnologije na kojoj je aplikacija implementirana.

Jedan od pristupa za ublažavanje ovog problema jeste uvođenje keširanja na pristupnicima, čime se može značajno smanjiti vreme pristupa sadržaju. Merenja pokazuju da se na ovaj način približno 76% zahteva može obraditi za manje od 250 ms [266].

Slično tome, određeno vreme je potrebno i da mreža utvrdi da traženi sadržaj ne postoji ni na jednom čvoru jer postoji određeno vreme potrebno za završetak procesa pretrage. Upit mora biti prosleđen između čvorova sve dok se ne pronađe odgovarajući sadržaj ili dok mreža ne zaključi da nijedan poznati čvor ne poseduje traženi CID, što predstavlja dodatni izvor vremena odaziva u odnosu na klasične centralizovane sisteme.

Dodatni izazov predstavlja trajnost podataka (eng. *data persistence*), jer IPFS ne garantuje da će jednom postavljena datoteka biti trajno dostupna, već njena dostupnost zavisi od toga da li je neki čvor i dalje čuva u svom lokalnom skladištu. Oslanjanje na usluge kačenja uvodi određeni stepen centralizacije, budući da dostupnost podataka postaje zavisna od treće strane [242]. Kao odgovor na ovaj problem razvijeni su decentralizovani sistemi za skladištenje podataka, koji koriste kriptoekonomske podsticaje kako bi učesnike motivisali na dugoročno čuvanje sadržaja.

Zbog otvorene prirode IPFS mreže i distribuiranog rutiranja zahteva, zlonamerni učesnici mogu pokušati da prate koji sadržaj korisnici pretražuju ili preuzimaju, što utiče na privatnost korisnika tokom pristupa sadržaju. Savremeni pristupi unapređenju privatnosti zasnivaju se na prikrivanju identiteta korisnika i/ili sadržaja tokom procesa pronalaženja podataka, čime se otežava povezivanje zahteva sa konkretnim učesnicima mreže [267], bez oslanjanja na pouzdane treće strane.

Sa aspekta privatnosti podataka, IPFS obezbeđuje integritet sadržaja putem kriptografskog heširanja, ali ne garantuje njegovu poverljivost. Svako ko poseduje odgovarajući CID može preuzeti sadržaj ukoliko je on dostupan u mreži. Iz tog razloga IPFS nije pogodan za skladištenje osetljivih podataka u izvornom obliku, već ih treba šifrovati pre postavljanja [268], a ključ za dešifrovanje čuvati odvojeno. Na ovaj način se zadržavaju prednosti sadržajno adresiranog skladištenja, uz istovremeno očuvanje poverljivosti podataka.

U tradicionalnim sistemima datoteke se mogu organizovati prema nazivima, direktorijumima ili metapodacima, što omogućava jednostavnu pretragu sadržaja na osnovu sličnih naziva ili drugih atributa. Sadržajno adresirani sistemi ne obezbeđuju mehanizme za semantičku pretragu sadržaja, zbog čega decentralizovane aplikacije moraju koristiti dodatne baze metapodataka ili indekse koji mapiraju korisnički razumljive identifikatore na odgovarajuće CID vrednosti [269].

Većina savremenih web pregledača ne podržava urođeni pristup IPFS sadržaju, zbog čega se u praksi često koriste pristupnici koji predstavljaju posrednika između korisnika i IPFS mreže. Slično tome, rezolucija ENS domena nije univerzalno podržana u svim pregledačima, pa je za pristup decentralizovanim aplikacijama često neophodno koristiti dodatne servise. Iako navedena ograničenja ne predstavljaju fundamentalne prepreke za funkcionisanje decentralizovanih aplikacija, ona pokazuju da potpuna decentralizacija prezentacionog sloja još uvek nije u potpunosti podržana od strane postojeće web infrastrukture.

Glava 9

Zaključak

Elektronska trgovina predstavlja jedan od najznačajnijih domena primene informacionih sistema, pri čemu savremena rešenja dominantno počivaju na centralizovanom modelu poverenja. U ovoj disertaciji analizirana je mogućnost primene javnih blokčejn sistema kao alternativne infrastrukturne osnove za realizaciju elektronske trgovine, sa posebnim fokusom na uticaj decentralizacije na arhitekturu sistema, troškove izvršavanja i korisničko iskustvo.

Sprovedena teorijska analiza pokazala je da javni blokčejn sistemi sa podrškom za pametne ugovore mogu zadovoljiti referentni model elektronske trgovine, ali uz uvođenje dodatne složenosti koja proizlazi iz potrebe za decentralizovanom koordinacijom i upravljanjem zajedničkim stanjem sistema. Potvrđeno je da decentralizacija predstavlja dodatnu dimenziju projektovanja informacionih sistema, koja utiče na raspodelu kontrole, odgovornosti i poverenja među učesnicima, kao i na arhitekturu infrastrukturnih i tehnoloških resursa sistema.

Na konstruktivnom nivou dizajniran je arhitektonski model platforme za elektronsku trgovinu zasnovan na decentralizovanim *Ethereum* i *IPFS* tehnologijama. Predloženi model omogućava realizaciju poslovne logike korišćenjem pametnih ugovora, skladištenja podataka i korisnički kontrolisanog kriptografskog identiteta, čime se uklanja potreba za centralnim autoritetom kao nosiocem poverenja.

Eksperimentalna analiza pokazala je da troškovi izvršavanja osnovnih operacija nad entitetima elektronske trgovine dominantno zavise od broja pristupa trajnom skladištu podataka. Na osnovu toga razvijen je optimizovani model reprezentacije entiteta unutar jedne skladišne jedinice, zasnovan na proširenju postojećih NFT standarda. Rezultati komparativne analize standarda ERC-721, ERC-721A i ERC-721Psi potvrdili su da se primenom optimizovanih modela skladištenja mogu ostvariti značajna smanjenja potrošnje gasa prilikom izvršavanja operacija kreiranja, prenosa i uništavanja entiteta.

Dobijeni rezultati potvrđuju opštu hipotezu istraživanja da je moguće projektovati distribuiranu platformu elektronske trgovine zasnovanu na javnoj blokčejn mreži koja primenom optimizovanih modela skladištenja i obrade podataka omogućava smanjenje naknade transakcionih troškova korisnika uz očuvanje funkcionalnih i bezbednosnih svojstava sistema. Ključni izazov nije realizacija funkcionalnosti sistema, već efikasno upravljanje ograničenim računarskim i skladišnim resursima decentralizovane infrastrukture.

Istraživanje je pokazalo da optimizacija modela reprezentacije podataka i pristupa skladištu predstavlja jedan od ključnih faktora za smanjenje operativnih troškova decentralizovanih sistema, bez narušavanja njihovih osnovnih funkcionalnih svojstava. Istovremeno, primena decentralizovanog

skladištenja podataka i prezentacionog sloja zasnovanih na IPFS mreži pokazuje da je moguće dodatno unaprediti stepen decentralizacije sistema uz očuvanje konzistentnosti i verifikabilnosti podataka.

Poseban značaj rezultata oglada se u formalizaciji veze između decentralizacije sistema, načina organizacije podataka i ekonomskih troškova izvršavanja poslovnih procesa. Na taj način uspostavljen je metodološki okvir koji omogućava sistematsku analizu i projektovanje decentralizovanih informacionih sistema ne samo u domenu elektronske trgovine, već i u drugim oblastima primene blokčejn tehnologija.

Naučni doprinos ove disertacije oglada se u formalizaciji referentnog modela elektronske trgovine u kontekstu javnih blokčejn sistema, definisanju analitičkog okvira za njihovu evaluaciju, razvoju optimizovanog modela reprezentacije entiteta unutar jedne skladišne jedinice, formalnom modelovanju mehanizama implicitnog vlasništva u NFT standardima, kao i eksperimentalnoj evaluaciji njihove troškovne efikasnosti.

Pored toga, predložen je arhitektonski model decentralizovane platforme za elektronsku trgovinu koji integriše javnu blokčejn mrežu, decentralizovano skladištenje podataka i decentralizovani prezentacioni sloj, čime je demonstrirana mogućnost realizacije potpuno decentralizovane arhitekture informacionog sistema elektronske trgovine.

Rezultati ovog istraživanja predstavljaju osnovu za dalji razvoj decentralizovanih informacionih sistema zasnovanih na javnim blokčejn mrežama. Buduća istraživanja mogu biti usmerena ka unapređenju interoperabilnosti između blokčejn sistema, optimizaciji performansi decentralizovanog skladištenja podataka, kao i razvoju novih modela za unapređenje privatnosti, skalabilnosti i efikasnosti decentralizovanih aplikacija.

Bibliografija

- [1] Salvador Carmona and Mahmoud Ezzamel. Accounting and accountability in ancient civilizations: Mesopotamia and ancient egypt. *Accounting, Auditing & Accountability Journal*, 20(2):177–209, 2007.
- [2] Ludwig Von Bertalanffy and John W Sutherland. General systems theory: Foundations, developments, applications. *IEEE Transactions on Systems, Man, and Cybernetics*, (6):592–592, 1974.
- [3] Eric L Trist et al. *The evolution of socio-technical systems*, volume 2. Ontario Quality of Working Life Centre Toronto, 1981.
- [4] Steven Alter. Defining information systems as work systems: implications for the is field. *European journal of information systems*, 17(5):448–469, 2008.
- [5] Steven Alter. A general, yet useful theory of information systems. *Communications of the association for information systems*, 1(1):13, 1999.
- [6] Steven Alter. Work system theory: overview of core concepts, extensions, and challenges for the future. *Journal of the Association for Information Systems*, page 72, 2013.
- [7] Steven Alter. The work system method: Systems thinking for business professionals. In *Proceedings of the 2012 Industrial and Systems Engineering Research Conference, Orlando, Florida*, 2011.
- [8] Mark H Moore. *Creating public value: Strategic management in government*. Harvard university press, 1997.
- [9] Vic Murray and Bill Tassie. Evaluating the effectiveness of nonprofit organizations. *The Jossey-Bass handbook of nonprofit leadership and management*, pages 303–324, 1994.
- [10] Kenneth C Laudon and Jane Price Laudon. *Management information systems: Managing the digital firm*. Pearson Educación, 2004.
- [11] Yuan Long, Keng Siau, and Min Ling. Towards a theoretical framework of information systems development success. *AMCIS*, 2005.
- [12] Pamela Zave. Classification of research efforts in requirements engineering. *ACM Computing Surveys (CSUR)*, 29(4):315–321, 1997.
- [13] Benjamin C Pierce. *Types and programming languages*. MIT press, 2002.

- [14] Alan Mathison Turing et al. On computable numbers, with an application to the entscheidungsproblem. *J. of Math.*, 58(345-363):5, 1936.
- [15] David T Bourgeois, James L Smith, Shouhong Wang, and Joseph Mortati. *Information systems for business and beyond*. Saylor Academy, 2019.
- [16] OECD. The 2025 oecd definition of e-commerce and guidelines for interpretation, 2025. OECD Publishing, Paris. <https://doi.org/10.1787/2254f1de-en>.
- [17] Prakash Veenam. Cross-border e-commerce: Opportunities and challenges in the global market. *International Journal of Multidisciplinary Research and Growth Evaluation*, 5(5):272–281, 2024.
- [18] Daniel A Menasc . A reference model for designing an e-commerce curriculum. *IEEE concurrency*, 8(1):82–85, 2000.
- [19] Aliaksandr Talaika, Joanna Biega, Antoine Amarilli, and Fabian M Suchanek. Harvesting entities from the web using unique identifiers–ibex. *arXiv preprint arXiv:1505.00841*, 2015.
- [20] Abbas N Albarq. Effect of web atmospherics and satisfaction on purchase behavior: stimulus–organism–response model. *Future Business Journal*, 7(1):62, 2021.
- [21] Ioannis Arapakis, Souneil Park, and Martin Pielot. Impact of response latency on user behaviour in mobile web search. In *Proceedings of the 2021 Conference on Human Information Interaction and Retrieval*, pages 279–283, 2021.
- [22] Andrew S Tanenbaum and Herbert Bos. *Modern operating systems*. Pearson Education, Inc., 2015.
- [23] Alessandra Levcovitz, Ricardo Terra, and Marco Tulio Valente. Towards a technique for extracting microservices from monolithic enterprise systems. *arXiv preprint arXiv:1605.03175*, 2016.
- [24] Juha Savolainen and Varvana Myllarniemi. Layered architecture revisited—comparison of research and practice. In *2009 Joint Working IEEE/IFIP Conference on Software Architecture & European Conference on Software Architecture*, pages 317–320. IEEE, 2009.
- [25] Laurent Michel and Pascal Van Hentenryck. A microkernel architecture for constraint programming. *Constraints*, 22(2):107–151, 2017.
- [26] Maarten Van Steen and Andrew S Tanenbaum. *Distributed systems*. Maarten van Steen Leiden, The Netherlands, 2017.
- [27] Joshua Bloch. How to design a good api and why it matters. In *Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications*, pages 506–507, 2006.
- [28] Andrew D Birrell and Bruce Jay Nelson. Implementing remote procedure calls. *ACM Transactions on Computer Systems (TOCS)*, 2(1):39–59, 1984.
- [29] Patrick Th Eugster, Pascal A Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. The many faces of publish/subscribe. *ACM computing surveys (CSUR)*, 35(2):114–131, 2003.

- [30] Ian Fette and Alexey Melnikov. The websocket protocol. Technical report, 2011.
- [31] Brenda M Michelson. Event-driven architecture overview. *Patricia Seybold Group*, 2(12):10–1571, 2006.
- [32] Nuha Alshuqayran, Nour Ali, and Roger Evans. A systematic mapping study in microservice architecture. In *2016 IEEE 9th international conference on service-oriented computing and applications (SOCA)*, pages 44–51. IEEE, 2016.
- [33] Peter Mell, Tim Grance, et al. The nist definition of cloud computing. *National institute of science and technology, special publication*, 800(2011):145, 2011.
- [34] Alex Berson. *Client/server architecture*. McGraw-Hill, Inc., 1992.
- [35] Eytan Bakshy, Solomon Messing, and Lada A Adamic. Exposure to ideologically diverse news and opinion on facebook. *Science*, 348(6239):1130–1132, 2015.
- [36] Sarah T Roberts. *Behind the screen*. Yale University Press, 2019.
- [37] Lina M Khan. Amazon’s antitrust paradox. *Yale LJ*, 126:710, 2016.
- [38] Shoshana Zuboff. The age of surveillance capitalism. In *Social theory re-wired*, pages 203–213. Routledge, 2023.
- [39] Alessandro Acquisti, Curtis Taylor, and Liad Wagman. The economics of privacy. *Journal of economic Literature*, 54(2):442–492, 2016.
- [40] Paul Baran. On distributed communications networks. *IEEE transactions on Communications Systems*, 12(1):1–9, 1964.
- [41] Vitalik Buterin. The meaning of decentralization, 2017. Pristupljeno: 2026-12-16. URL: <https://medium.com/@VitalikButerin/the-meaning-of-decentralization-a0c92b76a274>.
- [42] Andy Oram. *Peer-to-Peer: Harnessing the power of disruptive technologies*. "O’Reilly Media, Inc.", 2001.
- [43] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. In *Concurrency: the works of leslie lamport*, pages 203–226. 2019.
- [44] Al-Mukaddim Khan Pathan, Rajkumar Buyya, et al. A taxonomy and survey of content delivery networks. *Grid computing and distributed systems laboratory, University of Melbourne, Technical Report*, 4(2007):70, 2007.
- [45] Bram Cohen. Incentives build robustness in bittorrent. In *Workshop on Economics of Peer-to-Peer systems*, volume 6, pages 68–72, 2003.
- [46] Len Bass, Paul Clements, and Rick Kazman. *Software architecture in practice*. Addison-Wesley Professional, 2021.
- [47] Iso/iec 25010 : 2011 systems and software engineering - systems and software quality requirements and evaluation (square) - system and software quality models. 2013. URL: <https://api.semanticscholar.org/CorpusID:19022091>.

- [48] Eric A Brewer. Towards robust distributed systems. In *PODC*, volume 7, pages 343–477. Portland, OR, 2000.
- [49] Seth Gilbert and Nancy Lynch. Brewer’s conjecture and the feasibility of consistent, available, partition-tolerant web services. *Acm Sigact News*, 33(2):51–59, 2002.
- [50] OLGA VALERYEVNA Sedova and ALEXEY GRIGORIEVICH Alekseev. Development of mathematical models to determine the balance of the system of platform interactions when scaling the end-to-end monitoring process for priority sectors of the economy. *Journal of Theoretical and Applied Information Technology*, 101(1):11–20, 2023.
- [51] Raj Jain. *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. 1991.
- [52] Ivo Adan and Jacques Resing. *Queueing theory*, volume 180. Eindhoven, 2002.
- [53] André B Bondi. Characteristics of scalability and their impact on performance. In *Proceedings of the 2nd international workshop on Software and performance*, pages 195–203, 2000.
- [54] Mark D Hill and Michael R Marty. Amdahl’s law in the multicore era. *Computer*, 41(7):33–38, 2008.
- [55] Rong Ding and Haiying Du. Location-based ip addressing in ip-enable wireless sensor networks. In *2011 International Conference on Control, Automation and Systems Engineering (CASE)*, pages 1–4. IEEE, 2011.
- [56] Lukasz Kufel. Network latency in systems event monitoring for multiple locations. *Scientific Programming*, 2015(1):371620, 2015.
- [57] Neil J Gunther. A simple capacity model of massively parallel transaction systems. In *Int. CMG Conference*, pages 1–9, 1993.
- [58] Neil J Gunther. A general theory of computational scalability based on rational functions. *arXiv preprint arXiv:0808.1431*, 2008.
- [59] Heiko Hamann and Andreagiovanni Reina. Scalability in computing and robotics. *IEEE Transactions on Computers*, 71(6):1453–1465, 2021.
- [60] Martin Kleppmann. *Designing data-intensive applications: The big ideas behind reliable, scalable, and maintainable systems*. "O’Reilly Media, Inc.", 2017.
- [61] JoAnne Holliday, Robert Steinke, Divyakant Agrawal, and Amr El Abbadi. Epidemic algorithms for replicated databases. *IEEE Transactions on Knowledge and Data Engineering*, 15(5):1218–1238, 2003.
- [62] Jim Gray, Pat Helland, Patrick O’Neil, and Dennis Shasha. The dangers of replication and a solution. In *Proceedings of the 1996 ACM SIGMOD international Conference on Management of Data*, pages 173–182, 1996.
- [63] Carlo Curino, Evan Philip Charles Jones, Yang Zhang, and Samuel R Madden. Schism: a workload-driven approach to database replication and partitioning. 2010.

- [64] Weizhong Lu, Yuanchun Zhou, Kaichao Wu, and Baoping Yan. Hierarchical data replication and service monitoring methods in a scientific data grid. *Data Science Journal*, 8:113–124, 2009.
- [65] Timothy C. K. Chou and Jacob A. Abraham. Load balancing in distributed systems. *IEEE Transactions on Software Engineering*, (4):401–412, 2006.
- [66] Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. In *Concurrency: the Works of Leslie Lamport*, pages 179–196. 2019.
- [67] Philip A Bernstein, Vassos Hadzilacos, Nathan Goodman, et al. *Concurrency control and recovery in database systems*, volume 370. Addison-wesley Reading, 1987.
- [68] Jim Gray. Notes on data base operating systems. In *Operating Systems, An Advanced Course*, pages 393–481, Berlin, Heidelberg, 1978. Springer-Verlag.
- [69] Ngoc Thanh Nguyen. Consensus system for solving conflicts in distributed systems. *Information Sciences*, 147(1-4):91–122, 2002.
- [70] Leslie Lamport. The part-time parliament. In *Concurrency: the works of Leslie Lamport*, pages 277–317. 2019.
- [71] Miguel Castro, Barbara Liskov, et al. Practical byzantine fault tolerance. In *OsDI*, volume 99, pages 173–186, 1999.
- [72] Xavier Défago, André Schiper, and Péter Urbán. Total order broadcast and multicast algorithms: Taxonomy and survey. *ACM Computing Surveys (CSUR)*, 36(4):372–421, 2004.
- [73] Tushar Deepak Chandra and Sam Toueg. Unreliable failure detectors for reliable distributed systems. *Journal of the ACM (JACM)*, 43(2):225–267, 1996.
- [74] Guozhang Wang, Joel Koshy, Sriram Subramanian, Kartik Paramasivam, Mammad Zadeh, Neha Narkhede, Jun Rao, Jay Kreps, and Joe Stein. Building a replicated logging system with apache kafka. *Proceedings of the VLDB Endowment*, 8(12):1654–1655, 2015.
- [75] Fred B Schneider. Implementing fault-tolerant services using the state machine approach: A tutorial. *Acm Computing Surveys (CSUR)*, 22(4):299–319, 1990.
- [76] Michael J Fischer, Nancy A Lynch, and Michael S Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM (JACM)*, 32(2):374–382, 1985.
- [77] Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. Consensus in the presence of partial synchrony. *Journal of the ACM (JACM)*, 35(2):288–323, 1988.
- [78] Fabio Fagnani and Sandro Zampieri. Randomized consensus algorithms over large scale networks. *IEEE Journal on Selected Areas in Communications*, 26(4):634–649, 2008.
- [79] Michael Ben-Or. Another advantage of free choice (extended abstract) completely asynchronous agreement protocols. In *Proceedings of the second annual ACM symposium on Principles of distributed computing*, pages 27–30, 1983.
- [80] Matthew A. Bishop. *The Art and Science of Computer Security*. Addison-Wesley Longman Publishing Co., Inc., USA, 2002.

- [81] Florian Sommer, Jürgen Dürrewang, and Reiner Kriesten. Survey and classification of automotive security attacks. *Information*, 10(4):148, 2019.
- [82] Francesco Di Nocera and Giorgia Tempestini. Getting rid of the usability/security trade-off: A behavioral approach. *Journal of Cybersecurity and Privacy*, 2(2):245–256, 2022.
- [83] John Lynn Smith. *Design of Lucifer, a Cryptographic Device for Data Communications*. IBM Thomas J. Watson Research Center, 1971.
- [84] Horst Feistel. Cryptography and computer privacy. *Scientific american*, 228(5):15–23, 1973.
- [85] Joan Daemen and Vincent Rijmen. *The design of Rijndael*, volume 2. Springer, 2002.
- [86] Kazumaro Aoki, Tetsuya Ichikawa, Masayuki Kanda, Mitsuru Matsui, Shiho Moriai, Junko Nakajima, and Toshio Tokita. Camellia: A 128-bit block cipher suitable for multiple platforms—design and analysis. In *International workshop on selected areas in cryptography*, pages 39–56. Springer, 2000.
- [87] Daniel J Bernstein et al. Chacha, a variant of salsa20. In *Workshop record of SASC*, volume 8, pages 3–5. Lausanne, Switzerland, 2008.
- [88] Emilia Käsper and Peter Schwabe. Faster and timing-attack resistant aes-gcm. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 1–17. Springer, 2009.
- [89] Yoav Nir and Adam Langley. Chacha20 and poly1305 for ietf protocols. Technical report, 2018.
- [90] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976. doi:10.1109/TIT.1976.1055638.
- [91] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.
- [92] Avi Wigderson. P, np and mathematics—a computational complexity perspective. In *Proceedings of the International Congress of Mathematicians Madrid, August 22–30, 2006*, pages 665–712. European Mathematical Society-EMS-Publishing House GmbH, 2007.
- [93] Harald Helfgott. An improved sieve of eratosthenes. *Mathematics of Computation*, 89(321):333–350, 2020.
- [94] Richard Crandall and Carl Pomerance. *Prime numbers: a computational perspective*. Springer, 2005.
- [95] FIPS PUB. Digital signature standard (dss). *Fips pub*, pages 186–192, 2000.
- [96] Don Johnson, Alfred Menezes, and Scott Vanstone. The elliptic curve digital signature algorithm (ecdsa). *International journal of information security*, 1(1):36–63, 2001.
- [97] Minghua Qu. Sec 2: Recommended elliptic curve domain parameters. *Certicom Res., Mississauga, ON, Canada, Tech. Rep. SEC2-Ver-0.6*, 1999.
- [98] Daniel J Bernstein, Niels Duif, Tanja Lange, Peter Schwabe, and Bo-Yin Yang. High-speed high-security signatures. *Journal of cryptographic engineering*, 2(2):77–89, 2012.

- [99] Peter W Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134. Ieee, 1994.
- [100] Ralph C. Merkle. Protocols for public key cryptosystems. In *1980 IEEE Symposium on Security and Privacy*, pages 122–122, 1980. doi:10.1109/SP.1980.10006.
- [101] Lily Chen and Lily Chen. *Recommendation for key derivation using pseudorandom functions*. US Department of Commerce, National Institute of Standards and Technology, 2024.
- [102] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *International conference on the theory and applications of cryptographic techniques*, pages 453–474. Springer, 2001.
- [103] Eric Rescorla. The transport layer security (tls) protocol version 1.3. Technical report, 2018.
- [104] Fips Pub. Secure hash standard (shs). *Fips pub*, 180(4):2012, 2012.
- [105] Morris J Dworkin et al. Sha-3 standard: Permutation-based hash and extendable-output functions. 2015.
- [106] Hans Dobbertin, Antoon Bosselaers, and Bart Preneel. Ripemd-160: A strengthened version of ripemd. In *International Workshop on Fast Software Encryption*, pages 71–82. Springer, 1996.
- [107] Hugo Krawczyk, Mihir Bellare, and Ran Canetti. Hmac: Keyed-hashing for message authentication. Technical report, 1997.
- [108] Kim Cameron. The laws of identity. *Microsoft Corp*, 12(2005):8–11, 2005.
- [109] Christopher Metz. Aaa protocols: authentication, authorization, and accounting for the internet. *IEEE Internet Computing*, 3(6):75–79, 2002.
- [110] CTAM De Laat, George Gross, Leon Gommans, John Vollbrecht, and Dave Spence. Generic aaa architecture. Technical report, 2000.
- [111] Joseph Bonneau, Cormac Herley, Paul C Van Oorschot, and Frank Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *2012 IEEE symposium on security and privacy*, pages 553–567. IEEE, 2012.
- [112] Rachna Dhamija, J Doug Tygar, and Marti Hearst. Why phishing works. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 581–590, 2006.
- [113] Swarnima Rai, Vaaruni Choubey, Puneet Garg, et al. A systematic review of encryption and keylogging for computer system security. In *2022 fifth international conference on computational intelligence and communication technologies (CCICT)*, pages 157–163. IEEE, 2022.
- [114] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In *Annual international cryptology conference*, pages 232–249. Springer, 1993.
- [115] Dick Hardt. The oauth 2.0 authorization framework. Technical report, 2012.

- [116] Shafi Goldwasser, Silvio Micali, and Chales Rackoff. The knowledge complexity of interactive proof-systems. In *Providing sound foundations for cryptography: On the work of shafi goldwasser and silvio micali*, pages 203–225. 2019.
- [117] Aleksandr Ometov, Sergey Bezzateev, Niko Mäkitalo, Sergey Andreev, Tommi Mikkonen, and Yevgeni Koucheryavy. Multi-factor authentication: A survey. *Cryptography*, 2(1):1, 2018.
- [118] Butler W Lampson. Protection. *ACM SIGOPS Operating Systems Review*, 8(1):18–24, 1974.
- [119] Ravi S Sandhu and Pierangela Samarati. Access control: principle and practice. *IEEE communications magazine*, 32(9):40–48, 2002.
- [120] Niels Provos, Markus Friedl, and Peter Honeyman. Preventing privilege escalation. In *12th USENIX Security Symposium (USENIX Security 03)*, 2003.
- [121] Ebrima N Ceesay, Coimbatore Chandrasekaran, and William R Simpson. An authentication model for delegation, attribution and least privilege. In *Proceedings of the 3rd International Conference on Pervasive Technologies Related to Assistive Environments*, pages 1–7, 2010.
- [122] Bruce Schneier and John Kelsey. Secure audit logs to support computer forensics. *ACM Transactions on Information and System Security (TISSEC)*, 2(2):159–176, 1999.
- [123] Avijit Mallik. Man-in-the-middle-attack: Understanding in simple words. *International Journal of Data and Network Science*, 2019.
- [124] Paul Syverson. A taxonomy of replay attacks [cryptographic protocols]. In *Proceedings The Computer Security Foundations Workshop VII*, pages 187–191. IEEE, 1994.
- [125] Anuj Kumar Baitha and Smitha Vinod. Session hijacking and prevention technique. *Int. J. Eng. Technol*, 7(2.6):193–198, 2018.
- [126] Tim Rains. *Cybersecurity threats, malware trends, and strategies*. Packt Publishing, 2020.
- [127] Fatima Salahdine and Naima Kaabouch. Social engineering attacks: A survey. *Future internet*, 11(4):89, 2019.
- [128] Liu Hua Yeo and James Banfield. Human factors in electronic health records cybersecurity breach: an exploratory analysis. *Perspectives in health information management*, 19(2):li, 2022.
- [129] Adow Ali, Craig Overboe, Thivanka Dissanayake Mohottalalage, and Akalanka Mailewa. The human factor in cybersecurity: Understanding psychology, training efficacy, and error reduction strategies. 04 2024.
- [130] Felix Lau, Stuart H Rubin, Michael H Smith, and Ljiljana Trajkovic. Distributed denial of service attacks. In *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics. 'cybernetics evolving to systems, humans, organizations, and their complex interactions' (cat. no. 0*, volume 3, pages 2275–2280. IEEE, 2000.
- [131] Stuart Haber and W Scott Stornetta. How to time-stamp a digital document. In *Conference on the Theory and Application of Cryptography*, pages 437–455. Springer, 1990.

- [132] Dave Bayer, Stuart Haber, and W Scott Stornetta. Improving the efficiency and reliability of digital time-stamping. In *Sequences II: Methods in Communication, Security, and Computer Science*, pages 329–334. Springer, 1993.
- [133] H. Massias, X. Avila, and Jean-Jacques Quisquater. Design of a secure timestamping service with minimal trust requirement. 07 2002.
- [134] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *Annual International Cryptology Conference*, 1992. URL: <https://api.semanticscholar.org/CorpusID:9236061>.
- [135] Adam Back. Hashcash-a denial of service counter-measure. 2002.
- [136] David Chaum. Blind signatures for untraceable payments. In *Advances in Cryptology: Proceedings of Crypto 82*, pages 199–203. Springer, 1983.
- [137] David Chaum. Security without identification: Transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.
- [138] Nick Szabo. Bit gold. 2005. Pristupljeno: 2026-12-16. URL: https://bitcoinstan.io/prehistory/doc/1998_2.pdf.
- [139] Wei Dai. B-money, 1998. Pristupljeno: 2026-12-16. URL: https://bitcoinstan.io/prehistory/doc/1998_3.pdf.
- [140] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.
- [141] Michael Crosby, Pradan Pattanayak, Sanjeev Verma, Vignesh Kalyanaraman, et al. Blockchain technology: Beyond bitcoin. *Applied innovation*, 2(6-10):71, 2016.
- [142] Nicholas Stifter, Aljosha Judmayer, Philipp Schindler, Alexei Zamyatin, and Edgar Weippl. Agreement with satoshi–on the formalization of nakamoto consensus. *Cryptology ePrint Archive*, 2018.
- [143] Jaya Klara Brekke and Wassim Zuhair Alsindi. Cryptoeconomics. *Internet Policy Review*, 10(2):1–6, 2021.
- [144] Federico Matteo Benčić and Ivana Podnar Žarko. Distributed ledger technology: Blockchain compared to directed acyclic graph. In *2018 IEEE 38th international conference on distributed computing systems (ICDCS)*, pages 1569–1570. IEEE, 2018.
- [145] Hung Dang, Tien Tuan Anh Dinh, Dumitrel Loghin, Ee-Chien Chang, Qian Lin, and Beng Chin Ooi. Towards scaling blockchain systems via sharding. In *Proceedings of the 2019 international conference on management of data*, pages 123–140, 2019.
- [146] Cosimo Sguanci, Roberto Spatafora, and Andrea Mario Vergani. Layer 2 blockchain scaling: A survey. *arXiv preprint arXiv:2107.10881*, 2021.
- [147] Adam Back, Matt Corallo, Luke Dashjr, Mark Friedenbach, Gregory Maxwell, Andrew Miller, Andrew Poelstra, Jorge Timón, and Pieter Wuille. Enabling blockchain innovations with pegged sidechains. URL: <http://www.opensciencereview.com/papers/123/enablingblockchain-innovations-with-pegged-sidechains>, 72:201–224, 2014.

- [148] Siamak Solat, Philippe Calvez, and Farid Naït-Abdesselam. Permissioned vs. permissionless blockchain: How and why there is only one right choice. *J. Softw.*, 16(3):95–106, 2021.
- [149] Hoang Tam Vo, Ziyuan Wang, Dileban Karunamoorthy, John Wagner, Ermyas Abebe, and Mukesh Mohania. Internet of blockchains: Techniques and challenges ahead. In *2018 IEEE international conference on internet of things (iThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCoM) and IEEE smart data (SmartData)*, pages 1574–1581. IEEE, 2018.
- [150] Maurice Herlihy, Barbara Liskov, and Liuba Shrira. Cross-chain deals and adversarial commerce. *arXiv preprint arXiv:1905.09743*, 2019.
- [151] Rafael Belchior, André Vasconcelos, Sérgio Guerreiro, and Miguel Correia. A survey on blockchain interoperability: Past, present, and future trends. *Acm Computing Surveys (CSUR)*, 54(8):1–41, 2021.
- [152] Itay Tsabary, Matan Yechieli, Alex Manuskin, and Ittay Eyal. Mad-htlc: because htlc is crazy-cheap to attack. In *2021 IEEE symposium on security and privacy (SP)*, pages 1230–1248. IEEE, 2021.
- [153] Aradhita Mukherjee, Luca Olivieri, Nabendu Chaki, and Agostino Cortesi. Double-spending attacks in cross-blockchain ecosystems. *Blockchain: Research and Applications*, page 100378, 2025.
- [154] Mengya Zhang, Xiaokuan Zhang, Yinqian Zhang, and Zhiqiang Lin. Security of cross-chain bridges: Attack surfaces, defenses, and open problems. In *Proceedings of the 27th International Symposium on Research in Attacks, Intrusions and Defenses*, pages 298–316, 2024.
- [155] Qianlan Bai, Yuedong Xu, Zhijian Zhou, and Xin Wang. Determining blockchain transaction timing and fee with observable mempools. *arXiv preprint arXiv:2512.21923*, 2025.
- [156] Anatoly Yakovenko. Solana: A new architecture for a high performance blockchain v0. 8.13, 2018.
- [157] Ralph C Merkle. A digital signature based on a conventional encryption function. In *Conference on the theory and application of cryptographic techniques*, pages 369–378. Springer, 1987.
- [158] Caixiang Fan, Sara Ghaemi, Hamzeh Khazaei, and Petr Musilek. Performance evaluation of blockchain systems: A systematic survey. *Ieee Access*, 8:126927–126950, 2020.
- [159] Kyle Croman, Christian Decker, Ittay Eyal, Adem Efe Gencer, Ari Juels, Ahmed Kosba, Andrew Miller, Prateek Saxena, Elaine Shi, Emin Gün Sirer, et al. On scaling decentralized blockchains: (a position paper). In *International conference on financial cryptography and data security*, pages 106–125. Springer, 2016.
- [160] Sisi Zhou, Kuanching Li, Lijun Xiao, Jiahong Cai, Wei Liang, and Arcangelo Castiglione. A systematic review of consensus mechanisms in blockchain. *Mathematics*, 11(10):2248, 2023.
- [161] Yusuke Kaneko. Comparing electricity consumption per use of blockchain and generative ai. *IEEE Access*, 2025.

- [162] Limeng Zhang, Rui Zhou, Qing Liu, Chengfei Liu, and M Ali Babar. Transaction fee estimation in the bitcoin system. *arXiv preprint arXiv:2405.15293*, 2024.
- [163] Christian Decker and Roger Wattenhofer. Bitcoin transaction malleability and mtgox. In *European symposium on research in computer security*, pages 313–326. Springer, 2014.
- [164] Nick Szabo. Smart contracts, 1994. Pristupljeno: 2026-12-16. URL: <https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/L0Twinterschool2006/szabo.best.vwh.net/smart.contracts.html>.
- [165] Qin Wang and Shiping Chen. Account abstraction, analysed. *arXiv preprint arXiv:2309.00448*, 2023.
- [166] Vitalik Buterin, Yoav Weiss, Dror Tirosh, Shahaf Nacson, Alex Forshtat, Kristof Gazso, and Tjaden Hess. Erc-4337: Account abstraction using alt mempoo, 2021. Pristupljeno: 2026-03-27. URL: <https://eips.ethereum.org/EIPS/eip-4337>.
- [167] Yue Liu, Qinghua Lu, Liming Zhu, Hye-Young Paik, and Mark Staples. A systematic literature review on blockchain governance. *Journal of Systems and Software*, 197:111576, 2023.
- [168] Youssef El Faqir, Javier Arroyo, and Samer Hassan. An overview of decentralized autonomous organizations on the blockchain. In *Proceedings of the 16th international symposium on open collaboration*, pages 1–8, 2020.
- [169] Primavera De Filippi and Samer Hassan. Blockchain technology as a regulatory technology: From code is law to law is code. *arXiv preprint arXiv:1801.02507*, 2018.
- [170] Jan Werth, Mohammad Hajian Berenjestanaki, Hamid R Barzegar, Nabil El Ioini, and Claus Pahl. A review of blockchain platforms based on the scalability, security and decentralization trilemma. *ICEIS (1)*, pages 146–155, 2023.
- [171] Christina Ovezik, Dimitris Karakostas, and Aggelos Kiayias. Sok: A stratified approach to blockchain decentralization. In *International Conference on Financial Cryptography and Data Security*, pages 128–155. Springer, 2024.
- [172] Christina Ovezik, Dimitris Karakostas, Mary Milad, Daniel W Woods, and Aggelos Kiayias. Sok: measuring blockchain decentralization. In *International Conference on Applied Cryptography and Network Security*, pages 184–214. Springer, 2025.
- [173] Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.
- [174] Corrado Gini. Measurement of inequality of incomes. *The economic journal*, 31(121):124–125, 1921.
- [175] William G Shepherd. Concentration ratios. In *The New Palgrave Dictionary of Economics*, pages 1–3. Springer, 1987.
- [176] Stephen A Rhoades. The herfindahl-hirschman index. *Fed. Res. Bull.*, 79:188, 1993.
- [177] Adem Efe Gencer, Soumya Basu, Ittay Eyal, Robbert Van Renesse, and Emin Gün Sirer. Decentralization in bitcoin and ethereum networks. In *International conference on financial cryptography and data security*, pages 439–457. Springer, 2018.

- [178] Li Peng, Wei Feng, Zheng Yan, Yafeng Li, Xiaokang Zhou, and Shohei Shimizu. Privacy preservation in permissionless blockchain: A survey. *Digital Communications and Networks*, 7(3):295–307, 2021.
- [179] Ethan Buchman. *Tendermint: Byzantine fault tolerance in the age of blockchains*. PhD thesis, University of Guelph, 2016.
- [180] Salem Alqahtani and Murat Demirbas. Bottlenecks in blockchain consensus protocols. In *2021 IEEE International Conference on Omni-Layer Intelligent Systems (COINS)*, pages 1–8. IEEE, 2021.
- [181] Taishi Nakai, Akira Sakurai, Shiori Hironaka, and Kazuyuki Shudo. The blockchain trilemma described by a formula. In *2023 IEEE International Conference on Blockchain (Blockchain)*, pages 41–46. IEEE, 2023.
- [182] Dodo Khan, Low Tang Jung, and Manzoor Ahmed Hashmani. Systematic literature review of challenges in blockchain scalability. *Applied Sciences*, 11(20):9372, 2021.
- [183] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger, shanghai version efc5f9a. *Ethereum project yellow paper*, pages 1–42, 2025. Pristupljeno: 2026-01-22. URL: <https://ethereum.github.io/yellowpaper/paper.pdf>.
- [184] Ethereum Foundation. Ethereum developer resources, 2026. Pristupljeno: 2026-01-29. URL: <https://ethereum.org/developers>.
- [185] HashKey Capital. “is this time different for ethereum?” a closer look at its current landscape and challenges, 2025. Pristupljeno: 2026-01-21. URL: https://hashkey.capital/news/details43_405.html.
- [186] Matteo Vaccargiu, Sabrina Aufiero, Cheick Ba, Silvia Bartolucci, Richard Clegg, Daniel Gaziotin, Rumyana Neykova, Roberto Tonelli, and Giuseppe Destefanis. Mining a decade of event impacts on contributor dynamics in ethereum: A longitudinal study. In *2025 IEEE/ACM 22nd International Conference on Mining Software Repositories (MSR)*, pages 552–563. IEEE, 2025.
- [187] beaconcha. Ethereum metrics. Pristupljeno: 2026-04-04. URL: <https://beaconcha.in/charts>.
- [188] Dominic Grandjean, Lioba Heimbach, and Roger Wattenhofer. Ethereum proof-of-stake consensus layer: Participation and decentralization. In *International Conference on Financial Cryptography and Data Security*, pages 253–280. Springer, 2024.
- [189] mike, Francesco, dapplion, Mikhail, Aditya, Justin, lightclient, and Felix Lange. Eip-7251: Increase the max_effective_balance, 2023. Pristupljeno: 2026-04-04. URL: <https://eips.ethereum.org/EIPS/eip-7251>.
- [190] dapplion and Tim Beiko. Eip-7514: Add max epoch churn limit, 2023. Pristupljeno: 2026-04-05. URL: <https://eips.ethereum.org/EIPS/eip-7514>.
- [191] Li Li. Mitigating challenges in ethereum’s proof-of-stake consensus: Evaluating the impact of eigenlayer and lido. *arXiv preprint arXiv:2410.23422*, 2024.

- [192] Kaya Alpturer and S Matthew Weinberg. Optimal randao manipulation in ethereum. *arXiv preprint arXiv:2409.19883*, 2024.
- [193] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *International conference on the theory and application of cryptology and information security*, pages 514–532. Springer, 2001.
- [194] Vitalik Buterin and Virgil Griffith. Casper the friendly finality gadget. *arXiv preprint arXiv:1710.09437*, 2017.
- [195] Vitalik Buterin, Diego Hernandez, Thor Kamphofner, Khiem Pham, Zhi Qiao, Danny Ryan, Juhyeok Sin, Ying Wang, and Yan X Zhang. Combining ghost and casper. *arXiv preprint arXiv:2003.03052*, 2020.
- [196] Sergio Solmonte. An analysis of the ethereum proof of stake protocol.
- [197] Dahlia Malkhi and Pawel Szalachowski. Maximal extractable value (mev) protection on a dag. *arXiv preprint arXiv:2208.00940*, 2022.
- [198] Tereza Burianová, Martin Perešíni, and Ivan Homoliak. Secret leader election in ethereum pos: An empirical security analysis of whisk and homomorphic sortition under dos on the leader and censorship attacks. *arXiv preprint arXiv:2509.24955*, 2025.
- [199] Ulysse Pavloff. A game-theoretic approach to the study of blockchain’s robustness. *arXiv preprint arXiv:2411.19175*, 2024.
- [200] Michael Neuder, Daniel J Moroz, Rithvik Rao, and David C Parkes. Low-cost attacks on ethereum 2.0 by sub-1/3 stakeholders. *arXiv preprint arXiv:2102.02247*, 2021.
- [201] Joachim Neu, Ertem Nusret Tas, and David Tse. Two attacks on proof-of-stake ghost/ethereum. *arXiv preprint arXiv:2203.01315*, 2022.
- [202] Donald R Morrison. Patricia—practical algorithm to retrieve information coded in alphanumeric. *Journal of the ACM (JACM)*, 15(4):514–534, 1968.
- [203] Vitalik Buterin and Martin Swende. Eip-2929: Gas cost increases for state access opcodes, 2020. Pristupljeno: 2026-02-04. URL: <https://eips.ethereum.org/EIPS/eip-2929>.
- [204] Wei Tang. Eip-2200: Structured definitions for net gas metering, 2019. Pristupljeno: 2026-02-04. URL: <https://eips.ethereum.org/EIPS/eip-2200>.
- [205] Martin Holst Swende, Paweł Bylica, Alex Beregszaszi, and Andrei Maiboroda. Eip-3860: Limit and meter initcode, 2021. Pristupljeno: 2026-04-10. URL: <https://eips.ethereum.org/EIPS/eip-3860>.
- [206] Vitalik Buterin. Eip-1014: Skinny create2, 2018. Pristupljeno: 2026-04-06. URL: <https://eips.ethereum.org/EIPS/eip-1014>.
- [207] Faizan Khan, Istvan David, Daniel Varro, and Shane McIntosh. Code cloning in smart contracts on the ethereum platform: An extended replication study. *IEEE Transactions on Software Engineering*, 49(4):2006–2019, 2022.

- [208] Carlos Perez, Wei Han Ng, and Guillaume Ballet. Eip-8058: Contract bytecode deduplication discount, 2025. Pristupljeno: 2026-04-06. URL: <https://eips.ethereum.org/EIPS/eip-8058>.
- [209] Vitalik Buterin. Eip-7: Delegatecall, 2015. Pristupljeno: 2026-04-07. URL: <https://eips.ethereum.org/EIPS/eip-7>.
- [210] Jim Smith and Ravi Nair. *Virtual machines: versatile platforms for systems and processes*. Elsevier, 2005.
- [211] Alexey Akhunov and Moody Salem. Eip-1153: Transient storage opcodes, 2018. Pristupljeno: 2026-02-04. URL: <https://eips.ethereum.org/EIPS/eip-1153>.
- [212] Vitalik Buterin, Eric Conner, Rick Dudley, Matthew Slipper, Ian Norden, and Abdelhamid Bakhta. Eip-1559: Fee market change for eth 1.0 chain, 2019. Pristupljeno: 2026-02-05. URL: <https://eips.ethereum.org/EIPS/eip-1559>.
- [213] Sophia Gold, Parithosh Jayanthi, Toni Wahrstätter, Carl Beekhuizen, Ansgar Dietrichs, Dankrad Feist, Alex Stokes, Josh Rudolph, Giulio Rebuffo, Storm Slivkoff, and Kamil Chodola. Eip-7935: Set default gas limit to 60m, 2025. Pristupljeno: 2026-02-06. URL: <https://eips.ethereum.org/EIPS/eip-7935>.
- [214] Micah Zoltu. Eip-2718: Typed transaction envelope, 2020. Pristupljeno: 2026-02-08. URL: <https://eips.ethereum.org/EIPS/eip-2718>.
- [215] Vitalik Buterin and Martin Swende. Eip-2930: Optional access lists, 2020. Pristupljeno: 2026-02-08. URL: <https://eips.ethereum.org/EIPS/eip-2930>.
- [216] Vitalik Buterin, Dankrad Feist, Diederik Loerakker, George Kadianakis, Matt Garnett, Mofi Taiwo, and Ansgar Dietrichs. Eip-4844: Shard blob transactions, 2022. Pristupljeno: 2026-02-08. URL: <https://eips.ethereum.org/EIPS/eip-4844>.
- [217] Vitalik Buterin, Sam Wilson, Ansgar Dietrichs, and lightclient. Eip-7702: Set code for eoas, 2024. Pristupljeno: 2026-02-08. URL: <https://eips.ethereum.org/EIPS/eip-7702>.
- [218] Ting Chen, Xiaoqi Li, Xiapu Luo, and Xiaosong Zhang. Under-optimized smart contracts devour your money. In *2017 IEEE 24th international conference on software analysis, evolution and reengineering (SANER)*, pages 442–446. IEEE, 2017.
- [219] Alexey Akhunov, Eli Ben Sasson, Tom Brand, Louis Guthmann, and Avihu Levy. Eip-2028: Transaction data gas cost reduction, 2019. Pristupljeno: 2026-04-10. URL: <https://eips.ethereum.org/EIPS/eip-2028>.
- [220] Daniel Perez and Benjamin Livshits. Broken metre: Attacking resource metering in evm. *arXiv preprint arXiv:1909.07220*, 2019.
- [221] Vitalik Buterin and Martin Swende. Eip-3529: Reduction in refunds, 2021. Pristupljeno: 2026-02-10. URL: <https://eips.ethereum.org/EIPS/eip-3529>.
- [222] Lixuan Luo and Mariano P Consens. Understanding gastoken adoption in the ethereum blockchain. In *Proceedings of the 2022 5th International Conference on Blockchain Technology and Applications*, pages 63–69, 2022.

- [223] Meni Rosenfeld et al. Overview of colored coins. *White paper, bitcoil. co. il*, 41:94, 2012.
- [224] Kyungchan Ko, Taeyeol Jeong, Jongsoo Woo, and James Won-Ki Hong. Survey on blockchain-based non-fungible tokens: History, technologies, standards, and open challenges. *International Journal of Network Management*, 34(1):e2245, 2024.
- [225] William Entriken, Dieter Shirley, Jacob Evans, and Nastassia Sachs. Erc-721: Non-fungible token standard, 2018. Pristupljeno: 2026-02-10. URL: <https://eips.ethereum.org/EIPS/eip-721>.
- [226] Kai Ma, Jintao Huang, Ningyu He, Zhuo Wang, and Haoyu Wang. Sok: On the security of non-fungible tokens. *Blockchain: Research and Applications*, page 100268, 2025.
- [227] Vukasin S Tasic, Irena Tasic, and Mladen Veinovic. The cost-effective management of multiple e-coupons on public ethereum-based chains using nft standard with implicit ownership. *IEEE Access*, 2024.
- [228] Lodovica Marchesi, Michele Marchesi, Giuseppe Destefanis, Giulio Barabino, and Danilo Tigano. Design patterns for gas optimization in ethereum. In *2020 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, pages 9–15. IEEE, 2020.
- [229] Saeed Ranjbar Alvar, Mohammad Akbari, David Yue, and Yong Zhang. Nft-based data marketplace with digital watermarking. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4756–4767, 2023.
- [230] Azuki. Erc721a, 2022. Pristupljeno: 2026-02-13. URL: <https://www.erc721a.org/>.
- [231] CtorLab. Erc721psi, 2022. Pristupljeno: 2026-02-16. URL: <https://github.com/estarrionlvetch/ERC721Psi>.
- [232] Nicolaas Govert De Bruijn. A combinatorial problem. *Proceedings of the Section of Sciences of the Koninklijke Nederlandse Akademie van Wetenschappen te Amsterdam*, 49(7):758–764, 1946.
- [233] Charles E Leiserson, Harald Prokop, and Keith H Randall. Using de bruijn sequences to index a 1 in a computer word. *Available on the Internet from http://supertech.csail.mit.edu/papers.html*, 3(5), 1998.
- [234] Foundry organization. Foundry. a blazing fast, portable and modular toolkit for ethereum application development written in rust, 2021. Pristupljeno: 2026-02-21. URL: <https://github.com/foundry-rs/foundry>.
- [235] OpenZeppelin. Openzeppelin docs, 2021. Pristupljeno: 2026-02-21. URL: <https://docs.openzeppelin.com/contracts/4.x/erc721>.
- [236] Mathis Steichen, Beltran Fiz, Robert Norvill, Wazen Shbair, and Radu State. Blockchain-based, decentralized access control for ipfs. In *2018 Ieee international conference on internet of things (iThings) and ieee green computing and communications (GreenCom) and ieee cyber, physical and social computing (CPSCom) and ieee smart data (SmartData)*, pages 1499–1506. IEEE, 2018.

- [237] Douglas C Youvan. The internet’s achilles’ heel: Cloudflare’s frightening control and the risks of centralization. 2025.
- [238] Joel Olofsson and Oskar Rubensson. Decentralized web hosting: A comparative analysis of decentralized-and centralized web hosting, 2022.
- [239] Juan Benet. Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561*, 2014.
- [240] Hector Sanjuan, Samuli Poyhtari, Pedro Teixeira, and Ioannis Psaras. Merkle-crdts: Merkle-dags meet crdts. *arXiv preprint arXiv:2004.00107*, 2020.
- [241] Petar Maymounkov and David Mazières. Kademlia: A peer-to-peer information system based on the xor metric. In *International workshop on peer-to-peer systems*, pages 53–65. Springer, 2002.
- [242] Sid Lamichhane and Patrick Herbke. Verifiable decentralized ipfs cluster: unlocking trustworthy data permanency for off-chain storage. In *2024 6th Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS)*, pages 1–4. IEEE, 2024.
- [243] Juan Benet, Nicola Greco, et al. Filecoin: A decentralized storage network. *Protocol Labs*, 1:1–36, 2018.
- [244] Ben Fisch, Joseph Bonneau, Nicola Greco, and Juan Benet. Scaling proof-of-replication for filecoin mining. *Report. Protocol Labs Research*, 2018.
- [245] Tal Moran and Ilan Orlov. Simple proofs of space-time and rational proofs of storage. In *Annual International Cryptology Conference*, pages 381–409. Springer, 2019.
- [246] Adam J Levitin. Not your keys, not your coins: Unpriced credit risk in cryptocurrency. *Tex. L. Rev.*, 101:877, 2022.
- [247] Faizan Ahmed Syed and TarunKumar Rudrabhatla. A systematic literature review on evaluating the security of cold vs hot wallets in crypto storage. *Available at SSRN 5357834*, 2025.
- [248] Pengcheng Xia, Haoyu Wang, Zhou Yu, Xinyu Liu, Xiapu Luo, and Guoai Xu. Ethereum name service: the good, the bad, and the ugly. *arXiv preprint arXiv:2104.05185*, 2021.
- [249] Michèle Finck. *Blockchain regulation and governance in Europe*. Cambridge University Press, 2018.
- [250] Aaron Wright and Primavera De Filippi. Decentralized blockchain technology and the rise of lex cryptographia. *Available at SSRN 2580664*, 2015.
- [251] European Union Agency for Law Enforcement Cooperation (Europol). The evolving threat landscape: How encryption, proxies and ai are expanding cybercrime, 2026. URL: <https://www.europol.europa.eu/cms/sites/default/files/documents/IOCTA-2026.pdf>, doi: 10.2813/5737847.

- [252] Hamza Salem, Hadi Salloum, Manuel Mazzara, Nursultan Askarbekuly, Leonrad Johard, and Giancarlo Succi. Hidden risks: The centralization of nft metadata and what it means for the market. In *International Conference on Advanced Information Networking and Applications*, pages 154–162. Springer, 2025.
- [253] Simon Fremaux and William Schwab. Erc-4494: Permit for ERC-721 NFTs, 2021. Pristupljeno: 2026-02-25. URL: <https://eips.ethereum.org/EIPS/eip-4494>.
- [254] Zach Burks, James Morgan, Blaine Malone, and James Seibel. Erc-2981: NFT Royalty Standard, 2020. Pristupljeno: 2026-02-25. URL: <https://eips.ethereum.org/EIPS/eip-2981>.
- [255] Noama Fatima Samreen and Manar H Alalfi. Reentrancy vulnerability identification in Ethereum smart contracts. In *2020 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, pages 22–29. IEEE, 2020.
- [256] Zama. Confidential EVM Smart Contracts using Fully Homomorphic Encryption, 2024. <https://github.com/zama-ai/fhevm-solidity>.
- [257] Xiwei Xu, Cesare Pautasso, Liming Zhu, Qinghua Lu, and Ingo Weber. A pattern collection for blockchain-based applications. In *Proceedings of the 23rd European Conference on Pattern Languages of Programs*, pages 1–20, 2018.
- [258] Peter Murray, Nate Welch, and Joe Messerman. Erc-1167: Minimal proxy contract, 2018. Pristupljeno: 2026-02-23. URL: <https://eips.ethereum.org/EIPS/eip-1167>.
- [259] Sajad Meisami and William Edward Bodell III. A comprehensive survey of upgradeable smart contract patterns. *arXiv preprint arXiv:2304.03405*, 2023.
- [260] Santiago Palladino, Francisco Giordano, and Hadrien Croubois. Erc-1967: Proxy storage slots, 2019. Pristupljeno: 2026-02-23. URL: <https://eips.ethereum.org/EIPS/eip-1967>.
- [261] Nick Mudge. Erc-2535: Diamonds, multi-facet proxy, 2020. Pristupljeno: 2026-02-23. URL: <https://eips.ethereum.org/EIPS/eip-2535>.
- [262] Abdeljalil Benicche. A study of blockchain oracles. *arXiv preprint arXiv:2004.07140*, 2020.
- [263] Sanidhay Arora, Yingjiu Li, Yebo Feng, and Jiahua Xu. Secplf: secure protocols for loanable funds against oracle manipulation attacks. In *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*, pages 1394–1405, 2024.
- [264] Lorenz Breidenbach, Christian Cachin, Benedict Chan, Alex Coventry, Steve Ellis, Ari Juels, Farinaz Koushanfar, Andrew Miller, Brendan Magauran, Daniel Moroz, et al. Chainlink 2.0: Next steps in the evolution of decentralized oracle networks. *Chainlink Labs*, 1:1–136, 2021.
- [265] S Kireieva and A Yevheniev. *zkTLS: Enhancing Secure Communications with Zero-Knowledge Proofs*. PhD thesis, 2024.
- [266] Dennis Trautwein, Aravindh Raman, Gareth Tyson, Ignacio Castro, Will Scott, Moritz Schubotz, Bela Gipp, and Yiannis Psaras. Design and evaluation of ipfs: a storage layer for the decentralized web. In *Proceedings of the ACM SIGCOMM 2022 Conference*, pages 739–752, 2022.

- [267] Thomas Katsantas, Yannis Thomas, Christos Karapapas, and George Xylomenos. Enhancing ipfs privacy through triple hashing. In *2024 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6. IEEE, 2024.
- [268] Sushanth S Manakhari and Ajinkya P Jadhav. Enhancing data security with decentralized cloud storage: an ipfs approach. In *World Congress in Computer Science, Computer Engineering & Applied Computing*, pages 27–39. Springer, 2024.
- [269] Maxim V Shevarev and Stanislav V Suvorov. Dynamic content-oriented indexing and replication for high-performance storage and analysis of big data in the ipfs network. *Supercomputing Frontiers and Innovations*, 12(2):74–89, 2025.