RaiBlocks: Besplatna Distribuirana Kriptovalutna Mreža

Colin LeMahieu
clemahieu@gmail.com

***Sažetak*—Nedavno, velika potražnja i ograničena skalabilnost, povećale su prosečnu brzinu i cenu transakcije postojećih kriptovaluta, ostavljajući za sobom loše korisničko iskustvo. Predstavljamo Vam RaiBlocks, kriptovalutu sa novom blocklattice arhitekturom u kojoj svaki korisnik ima svoj blockchain, omogućujući gotovo instantne brzine transakcija i neograničenu skalabilnost. Svaki korisnik ima svoj blockchain, sto mu omogućuje da se asinhrono sinhronizuje sa ostatkom mreže, što rezultira brzim transakcijama sa minimalnim opterećenjem. Transakcije vode evidenciju o bilansu računa, a ne o količini transakcije, sto omogućuje agresivnu optimizaciju baze podataka bez ugrožavanja sigurnosti. Do danas, RaiBlocks mreža je obradila 4.2 miliona transakcija sa neoptimizovanom bazom podataka veličine 1.7GB. RaiBlock-ove besplatne i trenutne transakcije čine ga savršenim izborom za korisćenje.**

***Indeks Pojmova*—kriptovalute, blockchain, raiblocks, distribuirana glavna knjiga, digitalne transakcije**

I. Uvod

OD pojave Bitkoina 2009. godine, dešava se veliki pomak sa tradicionalnog platnog sistema, prema savremenom sistemu placanja osnovanim na kriptografiji, koji je stvorio mogućnost da novac cuvamo i razmjenjujemo na pouzdan i siguran način [1]. Da bi bila efikasna, valuta mora biti lako prenosiva, nepovratna, i imati male ili nepostojeće naknade u transakciji. Povećano vreme transakcija, velike naknade i upitna mrežna skalabilnost, nametnule su pitanje o praktičnosti Bitkiona kao svakodnevnu valutu.

U ovom dokumentu predstavljamo vam RaiBlocks, niskolatentnu kriptovalutu građenu na inovativnoj block-lattice strukturi podataka, koja omogućuje neograničenu skalabilnost i besplatne transakcije. RaiBlocks je po svom dizajnu jednostavan protokol sa jedinom svrhom da bude kritpovaluta visokih performansi. RaiBlocks protokol može da radi na hardveru male snage, što mu omogućuje da bude praktična, decentralizovana kriptovaluta za svakodnevnu upotrebu.

Statistike o kriptovalutama iznesene u ovom dokumentu su tačne na dan objave dokumenta.

II. Pozadina

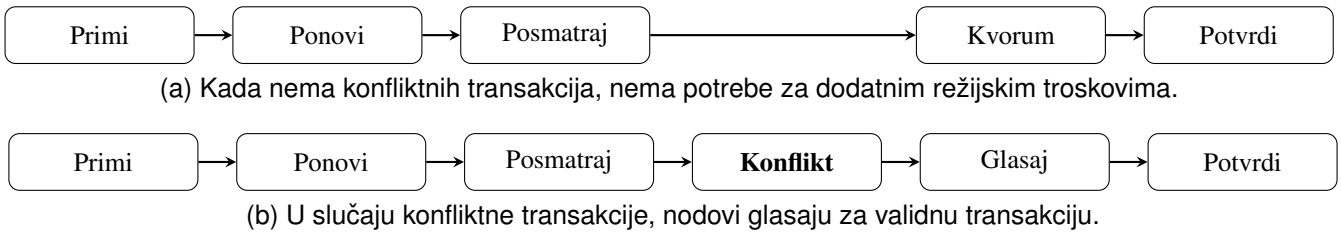
2008. godine, anonimni autor pod pseudonimom Satoši Nakamoto, objavio je dokument u kome je opisao prvu svetsku decentralizovanu kriptovalutu, Bitkoin [1]. Ključna inovacija u njegovom dokumentu, bio je blockchain, javna, nepromenljiva i decentralizovana struktura podataka koja se koristi kao knjiga bilansa za transakcije te valute. Nažalost, kako je Bitkoin sazrevao, nekoliko nedostataka u njegovom protokolu su ga napravile komplikovanim za mnogo aplikacija:

1) Slaba skalabilnost: Svaki blok u blockchainu može spremiti ogranicenu kolicinu podataka, što znači da sistem može obrađivati ograničenu količinu transakcija, što je stvorilo mesto u bloku, a time i samu transakciju skupom. Trenutno, prosečna cena transakcije je $10.38 [2].
2) Visoka latentnost: Prosečno vreme za konfirmaciju tranaskcije je 164 minuta. [3].
3) Neefikasna potrošnja struje: Bitkoin mreža u proseku konzumira 27.28TWh struje godišnje, trošeći prosecnih 260KWh po transakciji [4].

Bitkoin, i ostale kriptovalute, funkcionišu postizanjem konsenzusa o svojim globalnim knjigama bilansa da bi verifikovale legitimne transakcije, dok istovremeno onemogućuju lazne transakcije. Bitkoin postiže konsenzus ekonomskom merom zvanom Proof of Work (PoW). U PoW sistemu, učesnici se takmiče u računanju broja, nazvan *dokaz*, tako da je haš celoga bloka u ciljnom rasponu. Taj ciljni raspon je obrnuto proporcionalan kumulativnoj računalnoj snazi cele Bitkoin mreže, sa namerom da se dobije konzistentno prosečno vreme potrebno da se pronađe validan dokaz. Pronalazniku validnog dokaza je tada dozvoljeno da dodaje novi blok u blockchain; pa prema tome, oni koji potroše vise računarne snage na izračunavanje dokaza, igraju veću ulogu u stanju blockchaina. PoW omogućava otpornost na Sybil napade, gde se entitet jedinka ponaša kao više entiteta jedinki, da bi dobio veću moć u decentralizovanom sistemu, a takođe uveliko smanjuje uslove takmicenja koje postoji tokom pristupa globalnoj strukturi podataka.

Alternativni protokol saglasnosti, Proof of Stake (PoS), je prvi predstavio Peercoin 2012. godine [5]. U PoS sistemu, učesnici glasaju sa težinom koja je ekvivalentna količini bogatstva koje poseduju u datoj kriptovaluti. Tim aranžmanom, oni koji su više finansijski investirani, imaju veću moć koja im omogućuje da održavaju sigurnost sistema ili rizikuju gubitak svoje investicije. PoS na taj način zaobilazi potrebu za rasipnom računarskom snagom, i omogućuje upotrebu laganog softvera koji radi na energetski efikasnom hardveru.

Originalni RaiBlocks dokument i prva beta implementacija je objavljena 2014. godine, što je čini prvom Directed Acyclic Graph (DAG) kriptovalutom [6]. Ubrzo nakon toga, ostale DAG kriptovalute započele su svoj razvoj, među značajnijima su DagCoin/Byteball i IOTA [7], [8]. Navedene kriptovalute na osnovu DAG izašle su iz blockchain kalupa, unapredile performanse sistema i njegovu sigurnost. Byteball postiže konsenzus oslanjajući se na “main-chain” koji se sastoji od poštenih, reputabilnih i proverenih “witnesses”, dok IOTA



Slika 1. RaiBlocks ne zahteva nikakav dodatan rad u slučaju validne transakcije. U slučaju konfliktne transakcije, nodovi glasaju koju transakciju ce zadržati.

postiže konsenzus kumulativno dodajući POW transakcijama. RaiBlocks postiže dogovor uravnoteženom težinom saglasnosti kod konfliktnih transakcija. Takav sistem saglasnosti pruza brze i determinisane transakcije dok istovremeno zadržava snažan decentralizovani sistem. RaiBlocks nastavlja svoj razvoj i pozicionira se kao jedna od najizvrsnijih kriptovaluta.

III. RaiBlocks Komponente

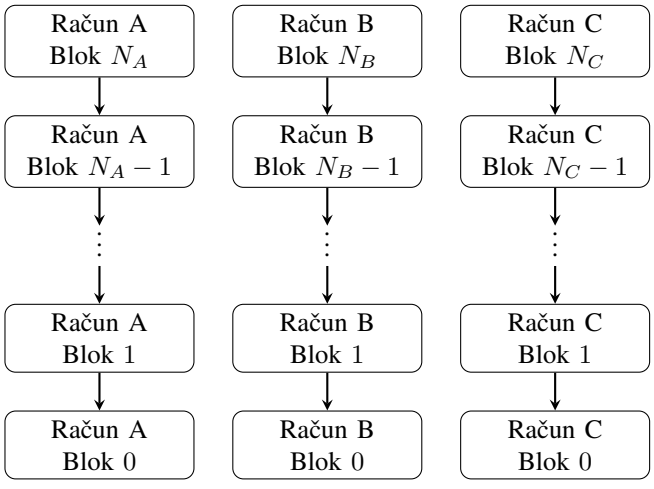
Pre nego što opišemo kompletnu RaiBlocks arhitekturu, ovde ćemo definisati individualne komponente koje čine njen sistem.

A. Račun

Račun je public-key deo digitalnog potpisa. Public-key, takođe korisćen kao adresa deli se sa ostalim učesnicima u mreži, dok je private-key tajan. Digitalno potpisani paket podataka brine se da je sadržaj bio odobren od vlasnika koji poseduje private-key. Jedan korisnik može kontrolisati nekoliko računa, ali samo jedna adresa može postojati po jednom računu.

B. Blok/Transakcija

Nazive "blok" i "transakcija" često koristimo naizmenično, gde blok sadrži jednu transakciju. Transakcija se specifično odnosi na akciju dok se blok odnosi na digitalno kodiranje transakcije. Transakcije potpisuje private-key koji pripada računu na kojem se transakcija izvršava.



Slika 2. Svaki račun ima svoj blockchain, koji sadrži bilans računa. Blok 0 mora biti otvorena transakcija. (Odeljak IV-B)

C. Knjiga bilansa

Knjiga bilansa je globalni set računa gde svaki račun ima svoju tablicu transakcija (Slika 2). Ovo je ključni komponent dizajna kojim menjamo run-time dogovor sa design-time dogovorom; svi saglasnici, proverom potpisa, slažu se da samo vlasnik moze modifikovati svoju tablicu transakcija. To pretvara naizgled deljenu strukturu podataka, podeljenu knjigu bilansa, u set nedeljenih knjiga bilansa.

D. Nod

Nod je računalni program koji se izvršava na računaru, usklađen je sa RaiBlocks protokolom i učestvuje u RaiBlocks mreži. Softver kontrolise glavnu knjigu i sve račune koje taj nod kontroliše, ako postoje. Nod može spremiti kompletnu knjigu bilansa, ili samo optimizovanu istoriju koja sadrži samo poslednjih nekoliko blokova svakog računa. Kod podešavanja novog noda preporuceno je verifikovati kompletnu istoriju i optimizirati je lokalno.

IV. Pregled Sistema

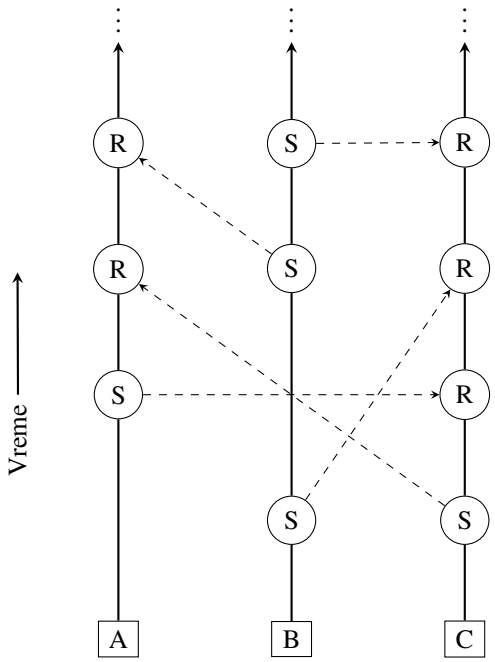
Za razliku od mnogih ostalih kriptovaluta koje koriste blockchain, RaiBlocks koristi *block-lattice* strukturu podataka. Svaki račun ima svoj blockchain (account-chain) koji je ekvivalentan istoriji transakcija/bilansa tog računa (Slika 2). Svaki račun može menjati samo njegov vlasnik; to omogućuje da račun bude menjan trenutno i asinhrono u odnosu na block-lattice, što rezultira brzim transakcijama. RaiBlocks protokol je izuzetno lagan; svaka transakcija stane u minimalni UDP paket koji se šalje internetom. Hardverski zahtevi za nodove su takođe minimalni, sa obzirom da nodovi za većinu transakcija samo moraju da vode evidenciju i salju blokove transakcija (Slika 1).

Sistem je iniciran sa *prvobitnim računom* koji sadrži *prvobitni bilans*. Prvobitni bilans je fiksni i nikada se nemože povećati. Prvobitni bilans je podeljen i poslan ostalim računima send transakcijom koja je registrovana u prvobitnom računu. Zbir svih bilansa svih računa nikada nece biti veći od inicijalnog prvobitnog računa, što sistemu daje granicu i onemogućuje mu da je poviši.

Ovaj odeljak objasniće kako su različiti tipovi transakcija izgrađeni i prosleđeni kroz mrežu.

A. Transakcije

Slanje sredstava sa jednog računa na drugi zahteva dve transakcije: *send* koja oduzima iznos sa računa pošiljaoca i *receive* koja dodaje iznos na račun primaoca (Slika 3).



Slika 3. Vizualizacija block-lattice. Svaki transfer novca zahteva send blok (S) i receive blok (R), svaki potpisan privatnim ključem njegovog vlasnika (A,B,C)

Slanje iznosa koristeći odvojene transakcije na računima pošiljaoca i primaoca ima nekoliko prednosti:

1) Nizanje ulaznih transfera koje su suštinski asinkroni.
2) Održavanje transakcija malima kako bi stale u UDP paket.
3) Olakšavanje optimizovanja glavne knjige, umanjujući njenu veličinu.
4) Odvajanje potvrđenih transakcija od nepotvrđenih.

Kada se više od jednog računa šalje prema jednom istom računu to je asinkrona operacija: latentnost mreže i poslati računi koji nisu neophodno u komunikaciji međusobno, znaci da ne postoji univerzalni dogovoreni način da znamo koja transakcija se dogodila prva. Sa obzirom da je dodavanje transakcija asocijativno, redosled kojim su ulazne transakcije poređane nije bitan, i potreban je globalni dogovor. Ovo je ključna komponenta dizajna koja pretvara run-time dogovor u design-time dogovor. Račun zato ima kontrolu da odluči koja je transakcija stigla prva i to čini potpisujući ulazne blokove.

Ako račun želi da napravi veliki transfer koji je primljen kao set nekoliko manjih transfera, to želimo da predstavimo na način koji stane u UDP paket. Kada račun primaoca prima ulazne transfere, konstantno pamti bilans svoga računa, tako da u svakom trenutku ima mogućnost da posalje bilo koji iznos fiksne veličine u transakciji. To je različito od ulazno/izlaznih tranaskcijskog modela koristenog u Bitkoin i ostalim kriptovalutama.

Neki nodovi nisu zainteresovani za trošenje resursa spremanjem kompletne istorije transakcija; zainteresovani su samo za trenutno stanje računa. Kada račun radi transakciju, on enkodira akumulirani bilans i prema tome nodovi treba da prate samo poslednji blok, što im dozvoljava da zanemare istoriju transakcija, dok istovremeo zadržavaju ispravnost.

Čak i sa fokusom na design-time sporazumima, postoji mogućnost kašnjenja prilikom validacije transakcija zbog identifikacije i obrade malicioznih aktera u mreži. Sa obzirom da se dogovori u RaiBlocks mreži rešavaju brzo, na nivou milisekunda do sekunde, korisniku možemo predstaviti dve slične kategorije ulaznih transakcija: rešene i nerešene transakcije. Rešene transakcije su transakcije za koje je račun primaoca generisao receive blok. Nerešene transakcije još nisu dodane u ukupan primaocev saldo. To je zamena za puno kompleksniji i nepoznatiji način potvrde transakcija u ostalim kriptovalutama.

B. Stvaranje Računa

Da bismo stvorili račun, potrebno je poslati *open* transakciju (Slika 4). Ta transakcija je uvek prva transakcija na svakom računu i može biti stvorena prilikom prvog primanja sredstava. Polje *account* sprema public-key (adresu) izvedenu iz private-key ključa, koji se koristi za potpisivanje. Polje *source* sadrži haš transakcije koja je poslala sredstva. Prilikom stvaranja računa potrebno je izabrati predstavnika, koji će glasati u ime stvorenog računa; predstavnik naknadno može biti promenjen (Odeljak IV-F). Račun može sam sebe proglasiti svojim predstavnikom.

```
open {
   account: DC04354B1...AE8FA2661B2,
   source: DC1E2B3F7C...182A0E26B4A,
   representative: xrb_1anr...posrs,
   work: 0000000000000000,
   type: open,
   signature: 83B0...006433265C7B204
}
```

Slika 4. Anatomija stvaranja transakcije

C. Saldo Račun

Saldo računa je spremljen unutar same knjige salda. Umesto zapisivanja kolicine iznosa transakcije, za verifikaciju (Odeljak IV-I) je potrebno proveriti razliku između salda send bloka i salda prethodnog bloka. Račun koji prima transakciju tada može povecćati prethodni saldo u finalni saldo koji je dat u novom primljenom bloku. To je učinjeno kako bi se poboljšala brzina obrade prilikom skidanja velike količine blokova. Kada zatražimo istoriju računa, iznosi su već poznati.

D. Slanje sa Računa

Da bismo poslali transakciju, adresa mora imati postojeći otvoreni blok, i prema tome saldo (Slika 5). Polje *previous* sadrži haš prethodnog bloka tog računa. Polje *destination* sadrži račun na koji šaljemo sredstva. Poslani blok je nepromenljiv jednom kada je potvrđen. Jednom kada je poslat u mrežu, sredstva su oduzeta sa poslaočevog računa i čekaju u polju *pending* dok primalac ne potpiše blok da bi prihvatio poslana sredstva. Sredstva koja su u toku, ne treba gledati kao na sredstva koja čekaju odobrenje, jer su potrošena sa pošiljaočevog računa i on ne može ponistiti transakciju.

```
send {
   previous: 1967EA355...F2F3E5BF801,
   balance: 010a8044a0...1d49289d88c,
   destination: xrb_3w...m37goeuufdp,
   work: 0000000000000000,
   type: send,
   signature: 83B0...006433265C7B204
}
```

Slika 5. Anatomija send transakcije

E. Receiving a Transaction

Da bismo završili transakciju, primalac mora da stvori blok na svom računu (Slika 6). Source polje pokazuje na haš pripadajuće poslate transakcije. Kad je taj blok kreiran i poslat u mrežu, saldo računa je osvežen i sredstva su zvanično primljena na njegov račun.

```
receive {
   previous: DC04354B1...AE8FA2661B2,
   source: DC1E2B3F7C6...182A0E26B4A,
   work: 0000000000000000,
   type: receive,
   signature: 83B0...006433265C7B204
}
```

Slika 6. Anatomija primanja transakcije

F. Dodeljivanje Predstavnika

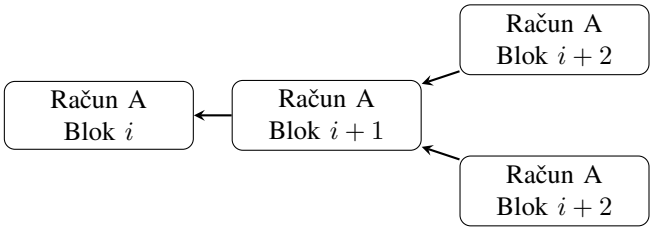
Mogucnost da vlasnici računa mogu izabrati predstavnika, koji ce glasati za njih je moćan decentralizacijski alat koji ne postoji u ostalim Proof of Work i Proof of Stake protokolima. U konvencionalnom PoS sistemu, nod vlasnika računa mora biti aktivan da bi učestvovao u glasanju. To je nepraktično za mnoge korisnike; mogućnost da korisnik ima predstavnika koji može glasati za njega rešava taj problem. Vlasnici računa imaju mogućnost promene predstavnika u bilo kojem trenutku. *Change* transakcija menja predstavnika računa, tako da oduzima moć glasanja sa starog predstavnika i daje je novom predstavniku (Slika 7). Za tu transakciju se ne troše sredstva, a predstavnik nema mogućnost trošenja sredstava računa koji ga je izabrao kao predstavnika.

```
change {
   previous: DC04354B1...AE8FA2661B2,
   representative: xrb_1anrz...posrs,
   work: 0000000000000000,
   type: change,
   signature: 83B0...006433265C7B204
}
```

Slika 7. Anatomija menjanja transakcije

G. Grananje i Glasanje

Do grananja dolazi kada j potpiše blokove $b_1, b_2, \ldots, b_j$ i traži pravo na isti blok kao njegov predhodnik (Slika 8). Ti blokovi stvaraju konflikt u statusu računa i moraju biti razrešeni. Samo vlasnik računa ima mogućnost potpisivanja blokova u svom računu, pa grananje može biti rezultat lošeg programiranja ili malicioznog pokušaja (double-spend) od strane vlasnika računa.



Slika 8. Do grananja dolazi kada dva (ili više) potpisanih blokova potraže isti prethodni blok. Stariji blokovi su na levoj strani, a noviji bokovi su na desnoj

U slučaju detekcije, predstavnik će pokrenuti glasanje koje pokazuje na blok $\hat{b}_i$ u svojoj knjizi salda i emitovaće ga u mrežu. Težina njegovog glasa, w_i, je zbir svih salda svih računa koji su ga izabrali kao predstavnika. Nod će tada pratiti glasove ostalih M aktivnih predstavnika i čuvati kumulativni zbir 4 glasačka perioda, u ukupnom trajanju 1 minuta, i potvrditi pobednički blok (Formula 1).

$$v(b_j) = \sum_{i=1}^{M} w_i \mathbb{1}_{\hat{b}_i = b_j} \tag{1}$$

$$b^* = \arg\max_{b_j} v(b_j) \tag{2}$$

Najpopularniji blok b^* će imati većinu glasova i biće zadržan u nodovoj knjizi salda (Formula 2). Blok koji izgubi glasanje je izbacen. Ako predstavnik zameni blok u svojoj glavnoj knjizi, to će stvoriti novo glasanje sa većim rednim brojem koje će biti emitovane u mrežu. Ovo je **jedini** scenario u kojem predstavnici glasuju.

U nekim okolnostima, kratke smetnje u mreži mogu prouzročiti da poslani blok nije prihvaćen od strane svih učesnika u mreži. Svaki sledeci blok na tom računu, koji učesnici nisu videli, ce biti ignorisan kao nevažeći. Ponovno slanje bloka će prihvatiti preostali učesnici a naknadni blokovi ce biti prihvaćeni automatski. Čak i kada se pojavi grananje ili nepostojeći blok, samo računi navedeni u transakciji su pogođeni; ostatak mreže nastavlja sa obradom transakcija za ostale račune.

H. Proof of Work

Sva četiri tipa transakcije imaju work polje koje mora biti ispravno popunjeno. Work polje omogućuje kreatoru transakcije da izračuna dokaz na nacin da je haš dokaza u vezi sa previous poljem u receive/send/change transakciji ili da je polje računa u otvorenoj transakciji ispod određene granične vrednosti. Za razliku od Bitkoin-a, PoW u RaiBlocks je jednostavno antispam alat, sličan Hashcash-u, i može biti izračunat

za nekoliko sekundi [9]. Kada je transakcija poslata, PoW za sledeci blok može biti unapred izračunat, sa obzirom da je prethodni blok poznat; na taj način transakcije deluju trenutne krajnjem korisniku, sve dok je vreme između transakcija veće od vremena potrebnog za računanje PoW.

I. Verifikacija Transakcije

Da bi se blok smatrao važećim, mora imati sledece atribute:

1) Blok ne sme postojati u knjizi salda (dupla transakcija).
2) Mora biti potpisan od strane vlasnika računa.
3) Prethodni blok je celni blok računa. Ako postoji, ali nije glavni, radi se o grananju.
4) Račun mora imati otvoreni blok.
5) Izračunati haš je unutar zahtevanog praga PoW granice.

U slučaju receive bloka, proverava se dali haš izvornog bloka čeka, što znači da još nije preuzet. Ako se radi o send bloku, saldo mora biti manji nego prethodni saldo.

V. Mogućnosti Napada

RaiBlocks, kao i sve decentralizovane kriptovalute, može biti napadnut od strane malicioznih korisnika, sa motivacijom financijske dobiti ili uništenja sistema. U ovom poglavlju objasni ćemo nekoliko mogućih scenarija napada, posledice takvih napada, i koje preventivne mere za njihovo sprečavanje poduzima RaiBlocks protokol.

A. Block Gap Synchronization

U Odeljku IV-G, razgovarali smo o scenariju u kojem blok nije primereno emitovan, što natera mrežu da ignoriše sledeće blokove. Ako nod vidi blok koji nema referencu prema prošlom bloku, ima dve opcije:

1) Ignorisati blok jer se moze raditi o malicioznom bloku.
2) Zatražiti resinhronizaciju sa drugim nodom.

U slucaju resinhronizacije, TCP veza mora biti formirana samopokrenutim nodom koji ce omogućiti povećanu količinu prometa koja je potrebna za resinhronizaciju. Međutim, ako je blok zaista bio loš blok, tada je resinhronizacija bila nepotrebna i stvaramo nepotreban promet na mreži. To je Network Amplification Attack koji rezultira napadom uskraćivanjem resursa (DoS napadom).

Da bismo izbegli nepotrebno resinhronizacije, nodovi ce čekati dok se promotri određen broj glasova za potencijalno maliciozni blok pre iniciranja resinhronizacije noda. Ako blok ne primi dovoljno glasova, možemo predpostaviti da je nod otpad.

B. Transaction Flooding

Maliciozni entitet bi mogao poslati puno nepotrebnih, ali validnih transakcija između računa koje kontroliše, sa namerom da zaguše mrežu. Sa obzirom na besplatne transakcije, napad može trajati neograničeno vreme. Svejedno, PoW potreban za svaku transakciju ograničava broj transakcija koje maliciozni entiteti mogu generisati bez značajnog invesitranja u računalne resurse. Čak i u slučaju takvog napada sa namerom da zaguši knjigu salda, nodovi koji ne prate istoriju svih transakcija imaju mogućnost optimizovanja i uklanjanja starih transakcija sa svoga računa; to u slučaju takvog napada onemogućuje korišćenje diskovnog prostora korisnika.

C. Sybil Attack

Entitet bi mogao stvoriti stotine Raiblock nodova na jednom računaru: iako, pošto je sistem glasanja zasnovan na težini saldo računa, dodavanje dodatnih nodova u mrežu napada neće osigurati dodatne glasove napadaču. Prema tome ne postoji prednost koju bi mogao dobiti koristeći Sybil attack.

D. Penny-Spend Attack

Penny-spend napad je napad u kome napadač šalje beskonačno male količine iznosa velikom broju računa sa namerom da trosi diskovni prostor nodova. Objavu blokova ograničava PoW, pa to do određene granice ograničava mogućnost kreiranja računa i stvaranja transakcija. Nodovi koji ne prate kompletnu istoriju transakcija mogu optimizovati račune gde sa statističkim merama možemo utvrditi račune koji nisu validni. Napokon, RaiBlocks je podešen da koristi minimalnu količinu prostora, pa je prostor potreban da se spremi dodatni račun, proporcionalan veličini open block$+$indexing $= 96\text{B} + 32\text{B} = 128\text{B}$. To znači da 1GB moze spremiti 8 miliona penny-spend računa. Ako nodovi žele optimizovati još agresivnije, mogu izračunati distribuciju osnovanu na pristupnoj frekvenciji i premestiti ređe koristene račune na sporiji diskovni prostor.

E. Precomputed PoW Attack

Sa obzirom da je vlasnik računa jedini entitet koji dodaje blokove u svoj račun, budući blokovi mogu biti unapred stvoreni, zajedno sa njihovim PoW, pre nego emitovanja u mrežu. Ovde napadač generiše mnostvo budućih blokova, svaki minimalne vrednosti, kroz duži period vremena. U određenom trenutku, napadač izvršava Denial of Service (DoS) napad zagušujuci mrežu sa mnostvom validnih transakcija, koje će ostali nodovi obraditi i proslediti u najbržem roku. Ovo je napredna verzija Transaction Flooding napada opisana u Odeljku V-B. Takav napad bi delovao samo kratko, ali bi mogao biti koristen zajedno sa ostalim napadima, kao >50% Attack (Odeljak V-F) da bi bio efikasniji. Ogrančenje broja transakcija i ostale tehnike se trenutno istražuju, da bi se onemogucili takvi napadi.

F. >50% Attack

Mera konsenzusa za RaiBlocks je sistem glasanja osnovan na težini glasa. Ako napadač uspe ostvariti više od 50% glasačke snage, moze poljuljati sistem glasanja i prouzrokovati pad sistema. Napadač moze da smanji količinu salda koji mora posedovati, onemogućujući vazecim nodovima glasanje DoS napadom. RaiBlocks preduzima sljedece mere da bi onemogucio takve napade:

1) Primarna odbrana od ovoga napada je ta što je tezina glasa vezana za investiciju u sistemu. Vlasnik računa je prinuđen održavati mrežu zdravom da bi zaštitio svoju investiciju. Pokušaj da ošteti mrežu bio bi destruktivan za celu mrežu, pa tako i za njegovu investiciju.
2) Cena ovog napada je proporcionalna tržišnoj vrednosti RaiBlocks-a. U PoW sistemu, moguće je smisliti tehnologiju koja daje neproporcionalnu kontrolu u odnosu na

financijsku investiciju, i ako je napad uspešan, ta tehnologija može biti ponovo koristena. Kod RailBlocks-a cena napada na sistem povećava se sa samim sistemom, i ako je napad uspešan, investicija u napad se ne može vratiti.

3) Da bi se održao maksimalan broj prisutnih glasača, sledeća linija obrane je glasanje predstavnika. Vlasnici računa koji nisu u mogućnosti pouzdano učestvovati u glasanju, mogu imenovati predstavnika koji može glasati za njih. Povećanjem broja i raznolikosti predstavnika povećava se otpornost mreže.
4) Grananja u RaiBlocks-u nikad nisu slučajna, nodovi imaju mogućnost donošenja odluke kako pristupati razgranatim blokovima. Jedini slučaj u kome su računi korisnika koji nije napadač ranjivi, je kada primaju sredstva sa računa napadača. Korisnik koji želi da se osigura, može da ceka manje ili više vremena pre nego što primi sredstva sa računa koji je generisao grananje, ili da odluči da nikad ne primi sredstva. Primaoci tako isto mogu da stvore odvojene račune za primanje novca sa sumnjivih računa i na taj nacin izoluju ostale račune.
5) Poslednja linija odbrane koja još nije implementirana je *block cementing*. Raiblocks je otišao jako daleko u tome da razreši grananje blokova brzo kroz glasanje. Nodovi mogu biti konfigurirani tako da cementiraju blokove, što bi im onemogućilo izmene nakon određenog vremena. Mreža je dovoljno sigurna kroz fokus na brzoj uspostavi transakcija koja onemogućuje grananja.

Nešto sofisticiranija verzija $> 50\%$ napada je objašnjena na Slici 9. “Offline” je procenat predstavnika koji su izabrani, ali nisu aktivni u glasanju. “Stake” je količina investicije sa kojom napadač glasa. “Active” je predstavnik koji je aktivan i glasa prema protokolu. Napadač može pomaknuti količinu iznosa koju može izgubiti tako da ostale predstavnike izbaci sa mreže koristeći DoS napad. Ako taj napad bude neprekidan, predstavnici neće biti sinhronizovani a to je prikazano sa “Unsync.” Napokon, napadač može ostvariti kratkoročne nalete u relativnoj snazi glasanja tako da prebacuje DoS napade na novi set predstavnika, dok se prethodno napadnuti predstavnici sinhronizuju, što je prikazano sa “Attack.”

Offline	Unsync	**Attack**	Active	Stake

Slika 9. Potencijalni dogovor glasanja koji smanjuje 51% napad

Ako je napadač sposoban da stvori situaciju u kojoj je Ulog >Aktivnosti kombinacijom navedenih napada, mogao bi uspesno da preokre glasove u knjizi salda po cenu svojeg uloga. Možemo proceniti koliko bi takav napad kostao razmatrajući tržišnu vrednost ostalih sistema. Ako procenimo da je 33% predstavnika offline ili napadnuto DoS napadom, napadač bi morao da kupi 33% tržišne vrednosti da bi mogao da izvrsi takav napad.

G. Bootstrap Poisoning

Što duze je napadač sposoban da drži stari privatni ključ sa bilansom, veća je verovatnoca da bilans koji je posedovao u to vreme neće imati aktivnog predstavnika zato što su se njegova salda i predstavnici preselili na druge račune. To znači da ako je nod samopokrenut na staru reprezentaciju mreže gde napadač drzi kvorum glasačkog uloga u odnosu na predstavnika u tom trenutku, mogli bi zaljuljati odluke glasanja za taj nod. Ako bi novi korisnik želeo interakciju sa bilo kim drugim osim sa nodom napadaca, sve njegove transakcije bi bile odbijene iz razloga što sadržavaju različitu glavu bloka. Zaključak je da nodovi mogu trošiti vreme novih nodova u mreži, davajući im netačne informacije. Da bismo to sprečili, nodovi mogu biti upareni sa inicijalnom bazom računa i provereno tačnih blokova; to je zamena za skidanje kompletne baze, sve do glavnog bloka. Što je baza bliža tome da bude aktuelna, veća je verovatnoca uspešne obrane od ovog napada. Na kraju, taj napad nije ništa gori od slanja netačnih podataka nodovima koji se samopokreću, sa obzirom da oni ne mogu da komuniciraju ni sa kim ko ima savremenu bazu podataka.

VI. Implementacija

Trenutno, opisana implementacija je izvedena u C++ programskom jeziku i izdaje nova izdanja od 2014. godine na Github-u [10].

A. Mogućnosti Dizajna

RaiBlocks arhitektura prati standarde opisane u ovom dokumentu. Sve dodatne implementacije su opisane ovde.

1) Algoritam Potpisa: RaiBlocks koristi modifikovani ED25519 algoritam eliptične krivulje sa Blake2b haširanjem za sve digitalne potpise [11]. ED25519 je izabran zbog brzine potpisivanja, brzine verifikacije i visoke sigurnosti.

2) Algoritam Haširanja: Sa obzirom da se algoritam haširanja koristi samo za kontrolu gušenja mreže, izbor algoritma je manje važan u odnosu na kriptovalute koje su bazirane na rudarenju. Naša implementacija koristi Blake2b za rezimiranje sadržaja blokova [12].

3) Funkcija za derivaciju ključa: U određenom novčaniku, ključevi su kriptirani koristeći lozinku koja ulazi u funkciju za derivaciju ključa i radi zaštite od ASIC napada. Trenutni pobednik je Argon2 [13] jedinog javnog nadmetanja sa ciljem kreiranja otporne funkcije za uklanjanje ključa.

4) Interval Bloka: Sa obzirom da svaki račun ima svoj blockchain, osvežavanja se mogu izvršavati asinkrono u odnosu na ostatak mreže. Prema tome ne postoje intervali bloka i transakcije mogu biti objavljene instantno.

5) UDP Protokol: Naš sistem je dizajniran da radi beskonačno koristeći minimalne količine računalnih resursa. Sve poruke u sistemu su dizajnirane da stanu u jedan UDP paket. To olakšava komunikaciju i učesće u mreži bez potrebe za uspostavljanje kratkorocne TCP veze. TCP se koristi samo za nove korisnike kada žele masovno samopokrenuti blockchain.

Nodovi mogu biti sigurni da su njihove poruke primljene u mrežu posmatrajuci komunikaciju transakcija drugih nodova jer bi trebale primiti nekoliko kopija natrag.

B. IPv6 i Multicast

Gradeći povrh UDP u buducnosti omogućuje implementaciju IPv6 multicast kao zamenu za tradicionalo transakcijsko gušenje i emitovanje glasova. To će smanjiti konzumaciju prometa na mreži i dati veću fleksibilnost nodovima.

C. Performanse

U trenutku pisanja, RaiBlocks mreža je obradila 4.2 miliona transakcija, što je dovelo do baze veličine 1.7GB. Brzina transakcije se meri u sekundama. Trenutna implementacija koja se izvršava na obicnom SSD-u, omogućava 10,000 transakcija u sekundi i jedino ograničenje je I/O tvrdoga diska.

VII. Korišćenje Resursa

Ovo je pregled koristenja resursa RaiBlocks noda. Dodatno, govorimo o idejama za smanjenje koristenja resursa u određenim situacijama. Ograničeni nodovi se tipicno zovu lagani nodovi, optimizovani ili simplified payment verification (SPV) nodovi.

A. Mreža

Količina mrežne aktivnosti, zavisi o tome koliko mreža doprinosi zdravlju mreže.

1) Predstavnik: Predstavnik nod zahteva maksimalnu količinu mrežnih resursa jer posmatra promet glasanja ostalih predstavnika i objavljuje svoje glasove.

2) Nepoverljiv: Nepoverljivi nod je sličan predstavniku, ali je samo posmatrač, ne poseduje privatni ključ predstavnika i samostalno ne objavljuje glasove.

3) Poverljiv: Poverljivi nod posmatra promet glasova jednog predstavnika kome veruje da će ispravno doneti odluku. To smanjuje količinu prometa jer predstavnik nema potrebe da komunicira sa nodom.

4) Lagan: Lagani nod je istovremeno i poverljivi nod koji posmatra promet na računima za koje je zainteresovan i minimalno opterećuje mrežu.

5) Samopokrenuti: Samopokrenuti nod služi deo, ili celu knjigu salda nodovima koji se spajaju na mrežu. To se radi koristeći TCP vezu umesto UDP jer su potrebne velike količine podataka što zahteva naprednije kontrole prometa.

B. Kapacitet Diska

U zavisnosti od potrebe korisnika, različite konfiguracije nodova imaju različite zahteve za diskovnim prostorom.

1) Istorijski: Nod zainteresovan za čuvanje kompletne istorije svih transakcija zahteva najvecu potrebu za diskovnim prostorom.

2) Trenutni: Zbog dizajna koji čuva akumulirana salda sa blokovima, nodovi moraju čuvati samo informaciju o poslednjoj glavi bloka za svaki račun ako žele učestvovati u glasanju.

3) Lagani: Lagani nod ne čuva nikakve informacije o knjizi salda i samo učestvuje u mreži na način da posmatra aktivnosti računa za koje je zainteresovan ili po potrebi stvara nove transakcije sa privatnim ključem koji poseduje.

C. Procesor

1) Generisanje Transakcije: Nod zainteresovan za kreiranje transakcije mora da prikaže Proof of Work da bi prošao kroz Raiblocks-ov prigušni mehanizam. Performanse različitog hardvera su prikazane u Appendix A.

2) Predstavnik: Predstavnik mora verifikovati potpise za blokove, glasove i takođe prikazati svoje potpise da bi učestvovao u odlukama. Količina procesorske snage potrebne predstavniku je značajno manja od one potrebne za generisanje transakcije i moze raditi i na jednom procesoru suvremenog računara.

3) Posmatrač: Posmatrač nod ne generise svoje glasove. Sa obzirom da je generisanje potpisa minimalno, procesorski zahtevi su gotovo identicni kao i za nod predstavnika.

VIII. Zaključak

U ovom dokumentu, predstavili smo vam kriptovalutu koja je sigurna, besplatnih transakcija, koja koristi neuobičajenu block-lattice strukturu i delegirano Proof of Stake glasanje. Mreža zahteva minimalne resurse, ne treba skupi hardver za rudarenje, i može da obrađuje veliki broj transakcija. Sve ovo je postignuto korišćenjem individualnih blockchain-ova za svaki korisnicki račun, eliminisajući probleme pristupa i neefikasnosti globalnih baza podataka. Identifikovali smo moguće vrste napada na sistem i predstavili na koji način je Railblocks otporan na te napade.

Dodatak A
PoW Hardware testovi

Kao što je već pomenuto, PoW u RaiBlocks služi da smanji neželjeni spam na mreži. Nasa implementacija pruža mogućnost ubrzanja korišćenjem OpenCL i kompatibilnih GPU-a. Tablica I pruža stvarne testove i upoređuje različiti hardver. Trenutno PoW granica je fiksna, a adaptivna granica moze biti implementisana sa napredkom procesorske snage prosečnog računara.

Tabela I
Hardverske PoW Performanse

Uređaj	Transakcije u sekundi
Nvidia Tesla V100 (AWS)	6.4
Nvidia Tesla P100 (Google,Cloud)	4.9
Nvidia Tesla K80 (Google,Cloud)	1.64
AMD RX 470 OC	1.59
Nvidia GTX 1060 3GB	1.25
Intel Core i7 4790K AVX2	0.33
Intel Core i7 4790K,WebAssembly (Firefox)	0.14
Google Cloud 4 vCores	0.14-0.16
ARM64 server 4 cores (Scaleway)	0.05-0.07

Zahvale

Zahvaljujemo Brian Pugh-u na stvaranju i formatiranje ovog dokumenta.

LITERATURA

[1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008. [Online]. Available: http://bitcoin.org/bitcoin.pdf

[2] “Bitcoin median transaction fee historical chart.” [Online]. Available: https://bitinfocharts.com/comparison/bitcoin-median_transaction_fee.html

[3] “Bitcoin average confirmation time.” [Online]. Available: https://blockchain.info/charts/avg-confirmation-time

[4] “Bitcoin energy consumption index.” [Online]. Available: https://digiconomist.net/bitcoin-energy-consumption

[5] S. King and S. Nadal, “Ppcoin: Peer-to-peer crypto-currency with proof-of-stake,” 2012. [Online]. Available: https://peercoin.net/assets/paper/peercoin-paper.pdf

[6] C. LeMahieu, “Raiblocks distributed ledger network,” 2014.

[7] Y. Ribero and D. Raissar, “Dagcoin whitepaper,” 2015.

[8] S. Popov, “The tangle,” 2016.

[9] A. Back, “Hashcash - a denial of service counter-measure,” 2002. [Online]. Available: http://www.hashcash.org/papers/hashcash.pdf

[10] C. LeMahieu, “Raiblocks,” 2014. [Online]. Available: https://github.com/clemahieu/raiblocks

[11] D. J. Bernstein, N. Duif, T. Lange, P. Shwabe, and B.-Y. Yang, “High-speed high-security signatures,” 2011. [Online]. Available: http://ed25519.cr.yp.to/ed25519-20110926.pdf

[12] J.-P. Aumasson, S. Neves, Z. Wilcox-O’Hearn, and C. Winnerlein, “Blake2: Simpler, smaller, fast as md5,” 2012. [Online]. Available: https://blake2.net/blake2.pdf

[13] A. Biryukov, D. Dinu, and D. Khovratovich, “Argon2: The memory-hard function for password hashing and other applications,” 2015. [Online]. Available: https://password-hashing.net/argon2-specs.pdf