RaiBlocks: Maksuton, hajautettu kryptovaluuttaverkko

Colin LeMahieu
clemahieu@gmail.com

***Tiivistelmä*—Viimeaikoina korkea kysyntä ja rajattu skaalautuvuus on lisännyt keskimääräisiä transaktioaikoja ja maksuja suosituissa kryptovaluutoissa, johtaen heikkoihin kokemuksiin. Tässä esittelemme RaiBlocks -kryptovaluutan, ennennäkemättömän lohkosäleikköarkkitehtuurin missä jokaisella tilillä on oma lohkoketju, joka toimittaa lähes välittömän transaktionopeuden ja rajattoman skaalautuvuuden. Jokaisella käyttäjällä on oma lohkoketju, joka voidaan päivittää asynkronisesti muuhun verkkoon, josta johtuen päästään nopeisiin transaktioihin mahdollisimman pienillä kustannuksilla. Transaktiot pitävät kirjaa tilien saldosta, ennemmin kuin transaktiosummista, mahdollistaen aggressiivisen tietokantakarsinnan riskeeraamatta turvallisuutta. Tähän päivään mennessä RaiBlocks -verkko on prosessoinut 4.2 miljoonaa transaktiota karsimattomalla tilikirjalla, jonka koko on vain 1.7GB. RaiBlocksin maksuton, sadasosasekunnin transaktionopeus tekee siitä johtavan kryptovaluutan kuluttajatransaktioihin.**

***Termiluettelo*—kryptovaluutta, lohkoketju, raiblocks, hajautettu tilikirja, digitaalinen, transaktiot**

I. Johdanto

BITCOININ toteuttamisesta lähtien, vuonna 2009, on ollut nähtävissä kasvava muutos pois perinteisistä, hallituksen takaamista valuutoista ja taloudellisista järjestelmistä kohti moderneja maksutapoja jotka perustuvat kryptovaluuttoihin, jotka tarjoavat mahdollisuuden tallettaa ja siirtää varoja luotettavasti ja turvallisesti [1]. Toimiakseen tehokkaasti, valuutan pitää olla helposti siirrettävä, peruuttamaton ja maksua ei pidä ottaa tai maksun pitää olla rajattu. Kasvaneet transaktioajat, suuret maksut ja kyseenalainen verkon skaalautuvuus ovat nostaneet kysymyksiä Bitcoinin käytettävyydestä jokapäiväisenä valuuttana.

Tässä julkaisussa esittelemme RaiBlocksin, pieniviiveisen kryptovaluutan joka on rakennettu innovatiivisen lohkosäleikön päälle ja joka tarjoaa rajattoman skaalautuvuuden ja maksuttomat siirrot. RaiBlocks on designiltaan yksinkertainen protokolla, jonka ainut tarkoitus on olla suorituskykytehokas kryptovaluutta. RaiBlocks protokolla voi pyöriä vähän energiaa vaativien laitteistojen päällä, mahdollistaen käytännöllisyyden, hajautetun kryptovaluutan jokapäiväiseen käyttöön.

Kryptovaluuttastatistiikat jotka tässä julkaisussa esitetään, ovat tarkkoja julkaisupäivänä.

II. Taustoja

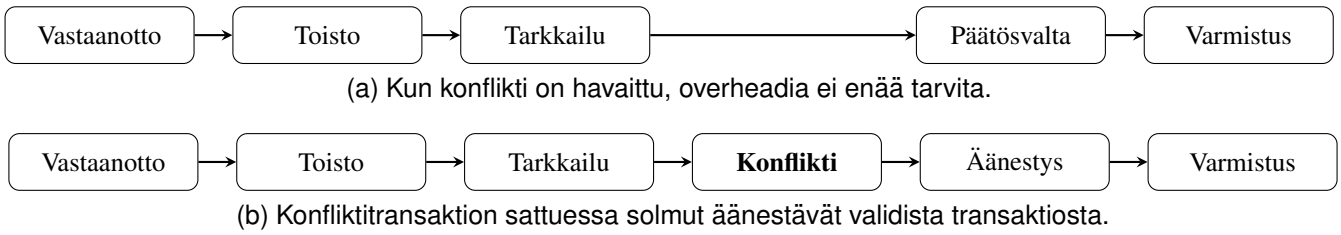
Vuonna 2008, tuntematon henkilö nimimerkiltä Satoshi Nakamoto julkaisi esitelmän jossa linjasi maailman ensimmäisen hajautetun kryptovaluutan, Bitcoinin [1]. Suurin innovaatio jonka Bitcoin loi, oli lohkoketju: julkinen muuttumaton ja hajautettu datarakenne, jota käytetään tilikirjana kryptovaluuttatransaktioihin. Valitettavasti, kun Bitcoin kypsyi, useat ongelmat protokollassa esti Bitcoinin käyttämisen monissa käyttökohteissa:

1) Huono skaalautuvuus: Jokainen lohko lohkoketjussa voi tallettaa vain rajoitetun määrän dataa, mikä tarkoittaa sitä, että järjestelmä voi prosessoida vain tietyn määrän transaktioita sekunnissa, tehden paikan lohkossa arvokkaaksi. Tällä hetkellä mediaanimaksu transaktiolle on $10.38 [2].
2) Korkea vasteaika: Keskimääräinen konfirmaatioaika on 164 minuuttia [3].
3) Tehoton virrankulutukselta: Bitcoin -verkko kuluttaa arviolta 27.28TWh vuodessa, käyttäen keskimäärin 260KWh transaktiota kohden [4].

Bitcoin, ja muut kryptovaluutat, toimivat saamalla vahvistuksen globaaleille tilikirjoille vahvistaakseen aidon transaktion samalla torjuen pahansuovat toimijat. Bitcoin saavuttaa konsensuksen ekonomisen toimenpiteen nimeltä Proof of Work (PoW) kautta. PoW -järjestelmässä osalliset kilpailevat laskeakseen numeron, jota kutsutaan termillä *nonce*, siten että koko lohkon tarkiste on kohdejoukossa. Tämä validi joukko on käänteisesti verrannollinen kumulatiiviseen Bitcoin -verkon laskentatehoon ylläpitääkseen keskimääräistä noncen löytöaikaa. Validin noncen löytäjä on oikeutettu lisäämään lohko lohkoketjuun; josta johtuen, ne ketkä antavat enemmän laskentatehoresursseja laskeakseen noncen, ovat tärkeämmässä roolissa lohkoketjun tilaan nähden. PoW tuo vastustuskykyä Sybli-hyökkäyksiä vastaan, missä entiteetti käyttäytyy kuin monta entiteettiä saadakseen lisää voimaa hajautetussa järjestelmässä, ja alentaakseen suuresti kilpailutilannetta joka luonnostaan on olemassa kun haetaan pääsyä globaaliin datarakenteeseen.

Vaihtoehtoinen konsensusprotokolla, Proof of Stake (PoS), esiteltiin ensimmäisenä Peercoinin toimesta vuonna 2012 [5]. PoS -järjestelmässä osallistujat äänestävät painolla, joka vastaa varallisuuttaan joka heillä on hallussa kyseisessä kryptovaluutassa. Tällä järjestelyllä, ne kellä on enemmän taloudellista panostusta saavat enemmän valtaa ja ovat luonnostaan taipuvaisia pitämään järjestelmän rehellisyyttä yllä tai riskeeraten sijoituksensa menettämisen. PoS poistaa hukkaavan laskentatehotarpeen, ja vaatii vain kevytrakenteisen ohjelmiston joka voi pyöriä vähäkulutuksellisella laitteistolla.

Alkuperäinen RaiBlocks -julkaisu ja ensimmäinen beta-implementaatio toteutettiin joulukuussa 2014, tehden siitä en-



Kuva 1. RaiBlocks ei vaadi ylimääräistä overheadia tyypilliselle transaktiolle. Konfliktitransaktiotilanteessa solmujen pitää äänestää pidettävästä transaktiosta.

simmäisen Direct Acyclic Graph (DAG) -pohjaisen kryptovaluutan [6]. Pian tämän jälkeen muita DAG kryptovaluuttoja alkoi kehittyä, etenkin DagCoin/Byteball ja IOTA [7], [8]. Nämä DAG -pohjaiset kryptovaluutat rikkoivat lohkoketjumuottia, parantaen järjestelmän suorituskykyä ja turvallisuutta. Byteball saavuttaa konsensuksen luottamalla "pääketjuun" joka muodostuu rehellisuudestä, hyvämaineisista ja käyttäjien luottamista "todistajista". IOTA puolestaan saavuttaa konsensuksen kumulatiivisen PoWin kautta joka perustuu pinottuihin transaktioihin. RaiBlocks saavuttaa konsensuksen tasepainotetun äänestyksen kautta kohdistuen konfliktoiviin transaktioihin. Tämä konsensusjärjestelmä tuottaa nopeamman, deterministisemmän transaktion pitäen yllä vahvaa hajautettua järjestelmää. RaiBlocks jatkaa tätä kehitystä ja on asettanut itsensä yhdeksi korkeasti suorituvimmista kryptovaluutoista.

III. RaiBlocksin Komponentit

Ennen kuin kuvaamme RaiBlocksin yleisarkkitehtuurin, määritämme yksittäiset komponentit joista järjestelmä rakentuu.

A. Tili

Tili on julkisavainosuus digitaalisen allekirjoituksen avainparista. Julkinen avain, jota kutsutaan myös osoitteeksi, on jaettu muiden verkon käyttäjien kanssa, kun taas salainen avain pidetään salaisena. Digitaalisesti allekirjoitettu datapaketti varmistaa, että sisältö on hyväksytty salaisen avaimen omistajan toimesta. Yksi käyttäjä voi kontrolloidn monia tilejä, mutta vain yksi julkinen osoite voi olla kohdistettu yhteen tiliin.

B. Lohko/Transaktio

Termiejä "lohko" ja "transaktio" käytetään usein liitännäisesti toisiinsa, missä lohko sisältää yhden transaktion. Transaktio viittaa erityisesti toimenpiteeseen, kun taas lohko viittaa transaktion digitaaliseen salaukseen. Transaktiot on allekirjoitettu salatulla avaimella joka kuuluu tilille miltä transaktio suoritetaan.

C. Tilikirja

Tilikirja on globaali joukko tilejä, missä jokaisella tilillä on oma transaktioketju (Kuva 2). Tämä on tärkeä suunnittelukomponentti joka kuuluu kategoriaan jossa korvataan ajonaikainen sopimus suunnittelunaikaisella sopimuksella; kaikki sopivat allekirjoituksen avulla että vain tilin omistaja voi muokata omaa ketjuaan. Tämä muuntaa näennäisesti jaetun datarakenteen, hajautetun tilikirjan, joukoksi jakamattomia.

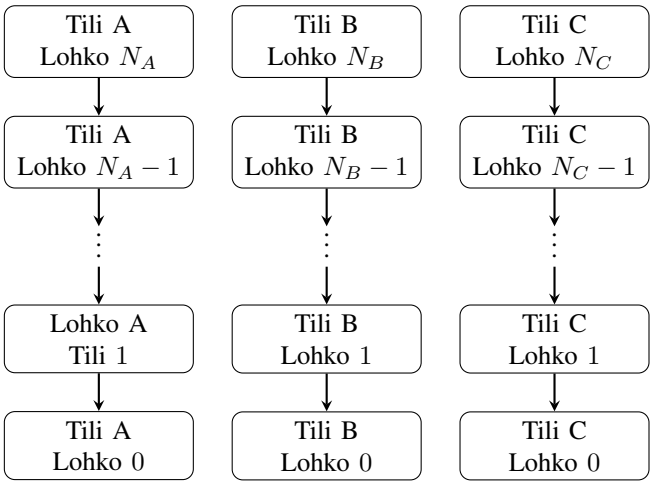
D. Solmu

Solmu on osa ohjelmistoa joka sovittaa RaiBlocks protokollan ja verkon toimijat yhteen RaiBlocks verkossa. Ohjelmisto hallinnoi tilikirjaa ja mitä tahansa tiliä, jos yhtäkään. Solmu voi tallettaa joko kokonaisen tilikirjan tai karsitun historian joka sisältää vain muutaman viimeisen lohkon jokaisen tilin lohkoketjusta. Kun uusi solmu perustetaan, on suositeltavaa verifioida koko historia ja karsia lokaalisti.

IV. Järjestelmän yleiskatsaus

Toisin kuin lohkoketjut joita käytetään muissa kryptovaluutoissa, RaiBlocks käyttää *lohkosäleikkö*rakennetta. Jokaisella tilillä on oma lohkoketjunsa (tiliketju) joka vastaa tilin transaktio- tai tasehistoriaa (Kuva 2). Jokaista tiliketjua voi päivittää vain tilin omistaja; tämä mahdollistaa jokaisen tiliketjun välittömän ja askynroonisen päivittämisen muuhun lohkosäleikköön, johtaen nopeaan transaktioon. RaiBlocksin protokolla on todella kevytrakenteinen; jokainen transaktio mahtuu pienimmän mahdollisen internetin yli lähetettävän UDP paketin kokoon. Laitteistovaatimukset solmuille ovat myös minimaaliset, koska solmujen pitää vain nauhoittaa ja lähettää lohkoja vain usealle transaktiolle (Kuva 1).

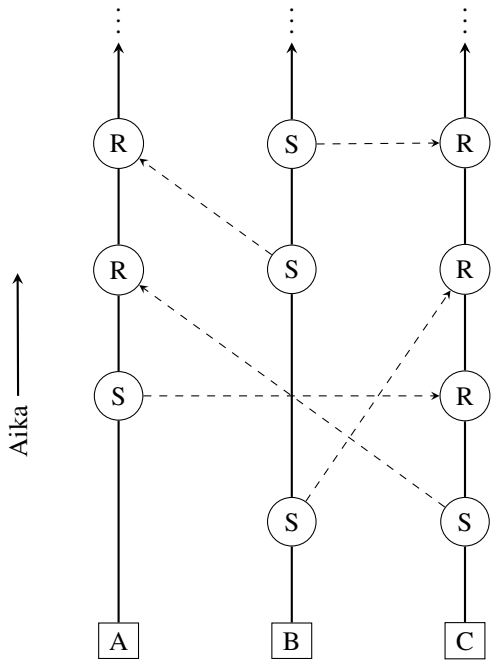
Järjestelmä alustetaan *alkutili*llä, joka pitää sisällään *alkutase*en. Alkutase on pysyvä määrä ja sitä ei voi koskaan lisätä. Alkutase jaetaan ja lähetetään alkutiliketjun muille tileille lähetystransaktion avulla, jotka on rekisteröity alkutiliketjuun. Kaikkien tilien taseiden summa ei koskaan ylitä al-



Kuva 2. Jokaisella tilillä on oma lohkoketjunsa sisältäen tilin tasehistorian. Lohko 0 pitää olla avoin transaktio (Kappale IV-B)

kuperäistä alkutasetta, joka määrää järjestelmälle suurimman mahdollisen voluumin ja estää tätä voluumia kasvamasta.

Tämä osio käy läpi miten erilaiset transaktiotyypit ovat rakentuneet ja propagoituneet verkon läpi.



Kuva 3. Lohkosäleikön havainnekuva. Jokainen varojen siirto vaatii lähettävän lohkon (S) ja vastaanottavan lohkon (R), jotka on allekirjoitettu oman tiliketjun omistajan toimesta (A,B,C)

A. Transaktiot

Varojen siirto tilillä toiselle vaatii kaksi transaktiota: *lähettävän*, joka vähentää lähettäjän tasetta, sekä *vastaanoton*, joka lisää summan vastaanottavan tilin taseeseen (Figure 3).

Summien lähettäminen erillisinä transaktioina lähettäjän ja vastaanottajan tileillä palvelee muutamaa tärkeätä tarkoitusta:

1) Saapuvien siirtojen ketjutus jotka ovat luonnostaan asynkronisia.
2) Transaktioiden pitäminen pienenä, jotta ne mahtuvat UDP pakettiin.
3) Helpottaa tilikirjan karsimista minimoimalla datan jalanjälkeä.
4) Eristämällä ratkaistut transaktiot ratkaisemattomista.

Useamman kuin yhden tilin lähettämistä samaan kohdetiliin kutsutaan asynkroniseksi operaatioksi; verkon vasteaika ja lähettävät tilit jotka eivät välttämättä kommunikoi keskenään tarkoittaa, että ei ole mitään universaalia tapaa selvittää mikä transaktio tapahtui ensiksi. Koska lisäys on assosiatiivista, syötteiden sekvensointijärjestys ei merkitse mitään, ja siksi tarvitsemme vain globaalin sopimuksen. Tämä on suunnittelun osalta tärkeä komponentti, joka muuntaa ajonaikaisen sopimuksen suunnitteluaikaiseksi sopimukseksi. Vastaanottavalla tilillä on valta päättää mikä siirto saapui ensimmäiseksi ja ilmaistaan järjestyksellä joka on saapuvien lohkojen signeerausjärjestys.

Jos tili haluaa tehdä suuren siirron joka on vastaanotettu pienten siirtojen kokonaisuutena, haluamme esittää tämän siten, että se mahtuu UDP paketin sisään. Kun vastaanottava tili ketjuttaa sisääntulevia siirtoja, se pitää juoksevaa kirjaa tilin kokonaissaldosta jotta koska tahansa on mahdollista siirtää mikä tahansa summa kiinteäkokoisena transaktiona. Tämä eroaa syöte/tuloste -transaktiomallista jota käyttää Bitcoin ja muut kryptovaluutat.

Osa solmuista ei ole kiinnostunut käyttämään resursseja tallettaaksen tilin täyttä transaktiohistoriaa; ne ovat kiinnostuneita vain jokaisen tilin senhetkisestä taseesta. Kun tili tekee transaktion, se koodaa kumuloidun taseen ja silloin kyseisten solmujen tarvitsee pitää kirjaa vain viimeisestä lohkosta. Tämä mahdollistaa solmuille historiallisen datan hylkäämisen säilyttäen kuitenkin datan oikeellisuuden.

Vaikka keskitytään suunnittelunaikaiseen sopimukseen, syntyy viivettä verkossa transaktioiden validoinnissa johtuen pahojen toimijoiden identifioinnista ja käsittelystä. Koska sopimukset RaiBlocksissa saavutetaan nopeasti, millisekunneista sekunteihin, voimme esittää käyttäjälle kaksi tuttua kategoriaa sisääntulevista transaktioista: ratkaistut ja ratkaisemattomat. Ratkaistut transaktiot ovat transaktioita jossa tili on luonut vastaanottavat lohkot. Ratkaisematon transaktio ei ole vielä korporoitunut vastaanottavan kumulatiiviseen taseeseen. Tämä on korvike kompleksisemmalle ja vieraammalle konfirmaatiometriikalle muissa kryptovaluutoissa.

B. Tilin luominen

Luodaksesi tilin, sinun pitää tehdä *avoin* transaktio (Kuva 4). Avoin transaktio on aina ensimmäinen transaktio jokaisessa tiliketjussa ja se voidaan luoda kun ensimmäisten varojen saapumisen yhteydessä. *Tili* -kenttä tallettaa julkisen avaimen (osoite) joka juonnetaan salaisesta avaimesta jota käytetään allekirjoittamiseen. *Lähde*kenttä sisältää transaktiotarkisteen siltä mikä lähetti varat. Tilin luomisen yhteydessä pitää valita edustaja joka äänestää puolestasi; tämän voi vaihtaa myöhemmin (Kappale IV-F). Tili voi julistautua omaksi edustajakseen.

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

Kuva 4. Avoimen transaktion anatomia

C. Tilin tase

Tilin tase on talletettu tilikirjan sisälle. Sen sijaan että talletettaisiin transaktion summa, verifikaatio (Kappale IV-I) vaatii tarkistuksen lähettävän ja vastaanottavan lohkon välillä.

Tämän jälkeen vastaanottava lohko voi lisätä edellisen lasketun taseen lopulliseen taseeseen joka on annettu vastaanotetussa lohkossa. Tämä tehdään jotta parannetaan prosessointinopeutta kun ladataan korkeatilavuuksisia lohkoja. Kun pyydetään tilihistoriaa, summat on jo annettu.

D. Tililtä lähettäminen

Lähettääkseen tililtä, osoiteella pitää olla jo olemassa oleva lohko ja siksi sillä on tase (Kuva 5). *Edellinen* kenttä sisältää sitä edellisen lohkon tarkisteen tiliketjussa. *Kohde*kenttä sisältää tilin jolle varat lähetetään. Lähetetty lohko on peruuttamaton kun se on varmistettu. Kun viesti on lähetetty verkkoon, varat ovat välittömästi vähennetty lähettäjän tililtä ja odottaa *vireillä* kunnes vastaanottava osapuoli allekirjoittaa lohkon hyväksyäkseen varat. Vireillä olevia varoja ei tulisi pitää sellaisena, että ne odottavat vahvistusta, koska ne ovat kulutettu lähettäjän tililtä ja lähettäjä ei voi peruuttaa transaktiota.

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

Kuva 5. Lähetystransaktion anatomia

E. Transaktion vastaanottaminen

Suorittaakseen transaktion loppuun, varojen vastaanottajan tulee luoda vastaanottolohko omaan tiliketjuunsa (Kuva 6). Lähdekenttä toimii tarkisteena assosioidulle lähetystransaktiolle. Kun tämä lohko on luotu ja lähetetty, tilin tase päivitetään ja varat siirretään virallisesti tilille.

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

Kuva 6. Vastaanottotransaktion anatomia

F. Edustajan määrittäminen

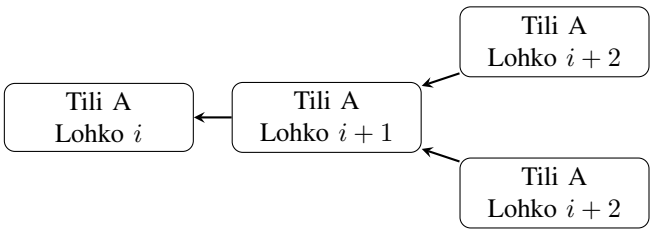
Tilien pitäjien mahdollisuus valita edustaja äänestämään heidän puolestaan on voikamas hajautustyökalu jolla ei ole vahvaa vertausta Proof of Work tai Proof os Stake -protokollissa. Perinteisissä PoS järjestelmissä tilin haltijan solmun pitää pyöriä jotta voi osallistua äänestykseen. Jatkuvasti solmun pyörittäminen on epäkäytännöllistä monille käyttäjille; antamalla edustajalle vallan äänestää tilin puolesta lieventää tätä vaatimusta. Tilien omistajat voivat vaihtaa edustajaansa mihin tahansa tiliin koska tahansa. Transaktion *muutos* muuttaa tilin edustajan vähentämällä äänipainon vanhan edustajan tililtä ja lisäämällä painon uudelle edustajalle (Kuva 7). Tässä transaktiossa ei siirretä varoja, ja edustalla ei ole oikeutta käyttää edustettavan tilin varoja.

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

Kuva 7. Muutostransaktion anatomia

G. Haarautuma ja Äänestys

Haarautuma tapahtuu kun j allekirjoittaneet lohkot $b_1, b_2, \ldots, b_j$ esittävät vaatimuksen saman lohkon olevan edeltäjälohko (Kuva 8). Nämä lohkot aiheuttavat konfliktoituneen näkymän tilin statukseen ja se pitää ratkaista. Vain tilin omistajalla on mahdollisuus allekirjoittaa lohkoja tiliketjuunsa, joten haarauman tulee olla tulos huonosta ohjelmoinnista tai pahansuovasta tarkoituksesta (tuplakulutus) tilin haltijan toimesta.



Kuva 8. Haarauma tapahtuu kun kaksi (tai useampi) allekirjoitettu lohko viittaa samaan edelliseen lohkoon. Vanhemmat lohkot vasemmalla; uudemmat lohkot oikealla

Havaittaessa, edustaja luo äänestyksen viitaten lohkoon $\hat{b}_i$ tilikirjassaan ja lähettää sen verkkoon. Solmun paino on kaikkien tilien taseiden summa jotka on nimenneet tilin edustajakseen. Tämä solmu tarkastelee sisääntulevia ääniä muilta verkon edustajilta ja pitää kumulatiivista kirjaa neljän äänestysjakson, 1 minuutin kokonaisuuden, ajan ja varmistaa voittavan lohkon (Kaava 1).

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

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

Suosituin lohko b^* tulee saamaan eniten ääniä ja säilyttää paikkansa solmun tilikirjassa (Kaava 2). Hävinnyt lohko hylätään. Jos edustaja korvaa lohkon tilikirjassaan, se luo uuden äänestyksen isommalla sekvenssinumerolla ja lähettää

tämän uudelleenäänestettäväksi verkkoon. Tämä on **ainoa** skenaario missä edustajat äänestävät.

Joissakin tapauksissa, hetkellinen verkkoyhteysongelma voi aiheuttaa lähetetyn lohkon osittaisen hylkäämisen. Mikä tahansa myöhempi lohko tällä tilillä sivuutetaan invalidina niiden vertaisten taholta, jotka eivät nähneet alkuperäistä lähetystä. Uudelleen lähetys tästä lohkosta hyväksytään jäljellä olevien vertaisten taholta ja jälkeentulevat lohkot haetaan automaattisesti. Vaikka haarauma tai puuttuva lohko tapahtuisikin, vain tilit jotka ovat mukana transaktiossa ovat vaikutuksen piirissä; loput verkosta jatkaa transaktioiden prosessoimista muulle verkolle.

H. Proof of Work

Kaikki neljä transaktiotyyppiä omaavat työkentän joka pitää täyttää korrektisti. Työkenttä mahdollistaa transaktion luojan laskea noncen jonka tarkiste on ketjutettu nonce edellisiin kenttiin vastaanotto/lähetys/muutos transaktioissa tai tili kentässä avoimessa transaktiossa on alle tietyn kynnyksen arvon. Toisin kuin Bitcoinissa, RaiBlocksin PoWia käytetään yksinkertaisesti anti-spam työkaluna, Hashcashin kaltaisesti, ja se voidaan laskea sekunneissa [9]. Kun transaktio on lähetetty, seuraava lohko voidaan esilaskea koska edellisen lohkon kenttä on tiedossa; tästä johtuen transaktio vaikuttaa välittömältä loppukäyttäjälle kunhan transaktioiden välinen aika on isompi kuin PoWin laskemiseen vaadittu aika.

I. Transaktion Varmistaminen

Jotta lohko voidaan katsoa validiksi, sen pitää omata seuraavat ominaisuudet:

1) Lohko ei saa olla jo tilikirjassa (duplikaatti transaktio).
2) Pitää olla allekirjoitettu omistajan toimesta.
3) Edellinen lohko on viimeisin lohko tiliketjussa. Jos lohko on olemassa, mutta se ei ole viimeinen, se on haarauma.
4) Tilillä pitää olla avoin lohko.
5) Laskettu tarkiste vastaa PoWin kynnysvaatimusta.

Jos se on lohkon vastaanotto, tarkista että lähdelohkon tarkiste on vireillä, tarkoittan että sitä ei ole vielä lunastettu. Jos se on lähetetty lohko, taseen pitää olla vähemmän kuin edellisen taseen.

V. Hyökkäysvektorit

RaiBlocks, kuten kaikki hajautetut kryptovaluutat, voi kohdata hyökkäyksiä pahansuovilta osapuolilta jotta ne saavuttaisivat taloudellista hyötyä tai järjestelmän alasajon. Tässä kappaleessa esitämme muutamia mahdollisia hyökkäystapoja, tapojen vaikutusta, sekä miten RaiBlocksin protokolla pyrkii estämään hyökkäykset.

A. Lohkovälin synkronointi

Kappaleessa IV-G kävimme läpi skenaarion missä lohkoa ei välttämättä kunnollisesti lähetetä, josta johtuen verkko hylkää jälkimmäiset lohkot. Jos solmu huomaa lohkon jolla ei ole referenssinä edellistä lohkoa, sillä on kaksi vaihtoehtoa:

1) Jättää lohko huomioitta koska se voisi olla pahansuopa roskalohko.
2) Pyytää uudelleensynkkausta toiselta solmulta.

Uudelleensynkkaystapauksessa TCP -yhteys pitää olla muodostettu esiladatun solmun kanssa jotta voidaan hoitaa kasvanut liikenne jonka uudelleensynkkaus tarvitsee. Kuitenkin, jos lohko oli oikeasti huono lohko, silloin uudelleensynkkaus oli tarpeeton ja se nosti tarpeettomasti liikennettä verkossa. Tämä on verkonvahvistushyökkäys (Network Amplification Attack) ja johtaa palvelunestoon (denial-of-service).

Välttääkseen tarpeetonta uudelleensynkkausta, solmu odottaa että tietty määrä ääniä on huomattu jotta tunnistettaisiin pahansuopa lohko ennen kuin käynnistetään yhteys esiladattuun solmuun synkronointia varten. Jos lohkolla ei ole tarpeeksi ääniä, sen voidaan olettaa olevan roskadataa.

B. Transaktioiden tukkiminen

Pahansuopa osapuoli voi lähettää monta tarpeetonta mutta validia transaktiota tilien välillä joita se kontrolloi saadakseen verkon kyllästettyä transaktioilla. Ilman transaktiomaksuja tätä voi jatkaa loputtomasti. Kuitenkin, vaadittu PoW jokaiselle transaktiolle rajoittaa transaktiomäärää jonka pahansuopa osapuoli voi luoda ilman että oleellisesti panostaa laskentatehoresursseihin. Vaikka tilikirja olisikin kyseenomaisen hyökkäyksen kohteena, solmut jotka eivät ole täysiä historiallisia solmuja voivat karsia vanhat transaktiot ketjustaan; tämä pienentää tilankäyttöä melkein kaikilta käyttäjiltä tämäntyylisissä hyökkäystilanteissa.

C. Sybil hyökkäys

Osapuoli voi luoda satoja RaiBlocks solmuja yhdelle koneelle; mutta koska äänestysjärjestelmä perustuu painotettuun tilin taseeseen, solmujen lisääminen ei lisää äänivaltaa hyökkääjälle. Sybilhyökkäyksistä ei siis saa mitään etua.

D. Pennikulutushyökkäys

Pennikulutushyökkäys on hyökkäys missä hyökkääjä käyttää rajattoman määrän pieniä summia isolle määrälle tilejä hukatakseen solmujen tallennustilaa. Lohkojulkaisu on määrärajattu PoWin toimesta, joten tämä rajoittaa tilien ja transaktioiden luontia tiettyyn pisteeseen asti. Solmut jotka eivät ole täysiä historiasolmuja voivat karsia tilejä jotka ovat tietyn statistiikka-arvon alapuolella, koska ne eivät todennäköisesti ole valideja tilejä. Lopulta, RaiBlocks on säädetty käyttämään pienintä pysyvää tallennustilaa, joten vaadittu tila yhden lisätilin säilyttämiseen on suhteellinen avoin lohko$+$indeksointi $= 96\text{B}+32\text{B} = 128\text{B}$ kokoon. Tämä vastaa sitä, että 1GB pystyy tallettamaan 8 miljoonaa pennikulutustiliä. Jos solmut haluavat karsia aggressiivisemmin, niiden pitää laskea jako perustuen pääsytaajuuteen ja delegoida harvemmin käytetyt tilit hitaampaan tallennuspaikkaan.

E. Esilaskettu PoW -hyökkäys

Koska tilin omistaja on ainoa osapuoli joka voi lisätä lohkoja tiliketjuun, peräkkäiset lohkot voidaan laskea niiden PoWin

kanssa ennen kuin ne lähetetään verkkoon. Tässä tilanteessa hyökkääjä generoi lukemattoman määrän peräkkäisiä lohkoja, kaikki mahdollisimman pienellä arvolla, pitkän aikajakson aikana. Tietyssä pisteessä hyökkääjä suorittaa palvelunestohyökkäyksen (DoS) tukkimalla verkon suurella määrällä valideja transaktioita joita muut solmut prosessoivat ja kaiuttavat mahdollisimman nopeasti. Tämä on paranneltu versio transaktiotukkimishyökkäyksestä joka kuvattiin kappaleessa V-B. Tälläinen hyökkäys toimisi vain hetkellisesti, mutta voitaisiin käyttää muiden hyökkäysten yhteydessä, kuten >50% hyökkäys (kappale V-F) jotta sen tehokkuus kasvaisi. Transaktion määrärajoitus ja muut tekniikat on tällä hetkellä tutkinnan alla, jotta tällaisten hyökkäysten vaikutusta voidaan rajoittaa.

F. >50% hyökkäys

Konsensuksen mittari RaiBlocksille on tasetasapainotettu äänestysjärjestelmä. Jos hyökkääjän on mahdollista saada yli 50% äänestysvahvuudesta, se voi aiheuttaa verkon konsensuksen vääristymän jolloin järjestelmä rikkoutuu. Hyökkääjä voi myös madaltaa tarvittua tasetta estämällä hyviä solmuja äänestämästä DoS hyökkäyksen avulla. RaiBlocks suojautuu tällaiselta hyökkäykseltä seuraavilla tavoilla:

1) Päätoiminen puolustus tämäntyylistä hyökkäystä vastaan on äänestyspainon sitominen sijoitukseen järjestelmässä. Tilinhaltijalle on insentiivisesti hyödykästä pitää yllä järjestelmän rehellisyyttä jotta voi suojata sijoituksiaan. Tilikirjan muokkaus olisi tuhoisaa järjestelmälle ja siksi hän menettäisi sijoituksensa.
2) Tämän hyökkäyksen hinta on suhteellinen RaiBlocksin markkina-arvoon. PoW järjestelmissä teknologiaa voidaan kehittää joka antaa epäsuhteellisen kontrollin verrattuna rahalliseen panostukseen ja jos hyökkäys on onnistunut tätä teknologiaa voidaan uudelleenkäyttää kun hyökkäys on valmis. RaiBlocksissa hyökkäyksen hinta skaalautuu järjestelmän itsensä kanssa, ja jos hyökkäys onnistuu, panostus hyökkäykseen ei ole takaisinsaatavissa.
3) Pitääkseen yllä suurinta äänestäjien päätäntävaltaa, on seuraava puolustustapa edustajaäänestys. Tilien omistajat, jotka eivät voi luotettavasti osallistua äänestämiseen yhteyssyistä voivat nimetä edustajan joka voi äänestää heidän taseensa arvolla. Edustajien ja diversiteetin lisääminen lisää verkon resilienssiä.
4) Haaraumat RaiBlocksissa eivät ole koskaan vahinkoja, joten solmut voivat tehdä päätöksen miten toimia haarautuneiden lohkojen kanssa. Hyökkäämättömät tilit ovat vaarassa vain silloin, kun niille lähetetään varoja hyökkäävältä tililtä. Tilit jotka haluavat olla turvassa lohkohaaraumilta voivat odottaa hetken pidempään ennenkuin vastaanottavat tililtä joka loi haaraumia tai eivät ota vastaan maksuja ollenkaan. Vastaanottajat voivat myös luoda erillisiä tilejä joita käyttävät kun vastaanottavat varoja epäilyttäviltä tileiltä eristääkseen maksut eri tilien väleillä.
5) Viimeinen puolustuslinja jota ei vielä ole implemntoitu on *lohkosementointi*. RaiBlocks tekee kaikkensa, jotta lohkohaaraumat ratkaistaan nopeasti äänestämällä. Solmut voidaan konfiguroida sementoimaan lohkot, joka estää niiden palauttamisen tietyn ajan jälkeen. Verkko on riittävän turvallinen nopean ratkaisemisen takia jotta voidaan estää tulkinnanvaraiset haaraumat.

Kehittyneempi versio $> 50\%$ hyökkäyksestä on selitetty tarkemmin kuvassa 9. “Offline” on se määrä edustajia jotka on nimetty, mutta eivät ole yhteydessä äänestääkseen. “Stake” on se määrä, jolla hyökkääjä on äänestämässä. “Active” on edustajien määrä jotka ovat yhteydessä ja äänestävät protokollan mukaan. Hyökkääjä voi muuttaa luovutettavan panoksensa määrää pudottamalla muita äänestäjiä yhteydestä DoS hyökkäyksellä. Jos tätä hyökkäystä voidaan ylläpitää, hyökättävät edustajat tulevat epäsynkronoiduiksi ja on esitetty termillä “Unsync.” Lopulta, hyökkääjä voi saada pienen lisäyksen relatiivista äänestysvoimaa vaihtamalla DoS hyökkäyksen kohdistumaan uusiin edustajiin kun vanhan hyökkäyksen kohteet vielä uudelleensynkronoivat tilikirjaansa. Tämä on esitetty termillä “Attack.”

Offline	Unsync	**Attack**	Active	Stake

Kuva 9. Mahdollinen äänestysjärjestley joka voisi vähentää 51% hyökkäysvaatimuksista.

Jos hyökkääjän on mahdollista aiheuttaa panoksen (Stake) olevan suurempi kuin aktiivisten (Active) yhdistämällä näitä tilanteita, olisi hyökkääjän mahdollista kääntää äänet tilikirjalla oman panoksensa kustannuksella. Voimme arvioida miten paljon tämäntyylinen hyökkäys voisi maksaa tutkimalla markkina-arvoa muilta järjestelmiltä. Jos arvioimme 33% edustajista olevan yhteydettömiä tai DoS -hyökkäyksen kohteena, hyökkäjä tarvitsisi ostaa 33% markkina-arvosta hyökätäkseen järjestelmää vastaan äänestämällä.

G. Esilatausmyrkytys

Mitä pidempään hyökkääjä pitää hallussaan vanhaa salaista avainta jolla on tasetta, sen todennäköisempää on, että senaikaisilla taseilla ei ole osallistuvaa edustajaa, koska niiden taseiden edustajat on siirtynyt uudemmille tileille. Tämä tarkoittaa sitä, että jos solmu on esiladattu vanhaan verkon edustajaan missä hyökkääjällä on majoriteetti äänivallassa muihin edustajiin nähden sinä aikana, hänellä olisi mahdollisuus oskilloida äänestyspäätöksiä siinä solmussa. Jos tämä uusi käyttäjä haluaisi olla tekemisissä kenen tahansa muun kuin hyökkäävän solmun kanssa, hänen kaikki transaktiot hylättäisiin koska niillä on eri viimeinen lohko. Lopputuloksena solmut voivat hukata uusien solmujen aikaa verkossa syöttämälle niille huonoa informaatiota. Jotta tämä voitaisiin estää, solmut voidaan parittaa alkuperäisen tilitietokannan kanssa ja hyväksihavaittujen viimeisten lohkojen kanssa; tämä on korvaus tietokannan lataamiselle aina esilohkoon asti. Mitä lähempänä lataus on nykytilannetta, sen suurempi todennäköisyys on hyökkäyksen estymiseksi. Lopulta tämä hyökkäys ei ole sen pahempi kuin roskadatan syöttäminen solmuille samalla kun esiladataan, koska ne eivät voisi transaktioida kenenkään kanssa jolla on nykyinen tietokanta.

VI. Implementaatio

Tällä hetkellä referenssi-implementaatio on toteutettu C++:lla ja on tuottanut julkaisuja vuodesta 2014 Githubiin [10].

A. Suunnitteluominaisuudet

RaiBlocksin toteutus pitää kiinni arkkitehtuuristandardista joka on määritelty tässä esityksessä. Lisämääritykset on kuvattu tässä.

1) Allekirjoitusalgoritmi: RaiBlocks käyttää muunneltua ED25519 elliptic curve algoritmia ja Blake2b tarkistusta kaikkiin digitaalisiin allekirjoituksiin [11]. ED25519 valittiin nopean allekirjoituksen, nopean verifioinnin ja korkean turvallisuuden takia.

2) Tarkistealgoritmi: Koska tarkistealgoritmia käytetään vain estämään verkon roskaa, algoritmivalinta on vähemmän tärkeä verrattuna louhintapohjaisiin algoritmeihin. Meidän implementaatio käyttää Blake2b:tä yhteenveto algoritminä lohkosisältöä vasten [12].

3) Avaimen deviointitoiminto: Referenssilompakossa avaimet ovat kryptattu salasanalla ja salasana on syötetty avaimen deviointitoiminnon läpi jotta se olisi suojassa ASIC murtoyrityksiltä. Tällä hetkellä Argon2 [13] on voittaja ja ainoa julkinen kilpailija joka tähtää luomaan resilientin avaimen deviointitoiminnon.

4) Lohkointervalli: Koska jokainen tili on oma lohkoketjunsa, päivitykset voidaan tehdä asynkronisesti verkon tilaan. Siksi ei ole lohkointervalleja ja transaktiot voidaan julkaista välittömästi.

5) UDP viestiprotokolla: Järjestelmämme on suunniteltu toimimaan loputtomasti käyttäen pienintä mahdollista laskentatehoa. Kaikki viestit järjestelmässä on suunniteltu tilattomiksi ja mahtumaan yhden UDP paketin sisään. Tämä tekee myös helpommaksi sen, että pienet toimijat rajatulla yhteydellä voivat osallista verkkoon ilman lyhytaikaisia TCP yhteyksiä. TCP on käytetty vain uusille toimijoille kun ne haluavat esilaskea lohkoketjuja bulkkityyliin.

Solmut voivat olla varmoja transaktioidensa vastaanotosta verkon toimesta seuraamalla transaktiolähetysliikennettä toisilta solmuilta, koska solmun pitäisi saada useiden kopioiden kaiku takaisin itselleen.

B. IPv6 ja Multicast

Yhteydettömän UDP:n päälle rakentaminen mahdollistaa IPv6 multicastin käytön uudemmissa implementaatioissa. Tämä korvaa perinteisen transaktioeston ja äänilähetyksen. Tämä vähentää verkon kaistanleveyden kulutusta ja antaa solmuille joustavuutta käytäntöjen kanssa.

C. Suorituskyky

Tämän kirjoitushetkellä, 4,2 miljoonaa transaktiota on prosessoitu RaiBlocks verkossa, joka vastaa 1,7GB lohkoketjukokoa. Transaktioajat on mitattu sekunneissa. Tämänhetkinen referenssitoteutus joka pyörii yleisillä SSD:illä voi prosessoida yli 10 000 transaktiota sekunnissa jotka on sidottu pääsääntöisesti IO:n.

VII. Resurssien käyttö

Tämä on yleiskatsaus resursseihin joita RaiBlocksin solmu käyttää. Lisäksi käymme läpi ideat joilla voidaan vähentää resurssien käyttöä tietyissä tilanteissa. Supistettuja solmuja kutsutaan tyypillisesti keveiksi, karsituiksi tai yksinkertaistetuksi maksuverifiointisolmuiksi (SPV, Simplified Payment Verification).

A. Verkko

Verkossa tapahtuvan liikenteen määrä riippuu siitä kuinka paljon verkko myötävaikuttaa verkon terveyteen.

1) Edustaja: Edustajasolmu vaatii maksimaalisen verkkoresurssin koska se tarkkailee äänestysliikennettä muilta edustajilta ja julkaisee omia ääniään.

2) Luottamaton: Luottamaton solmu on samanlainen kuin edustajasolmu, mutta on vain tarkkailija. Se ei sisällä edustajan tilin salaista avainta ja ei voi julkaista ääniä itsellään.

3) Luottava: Luottava solmu tarkkailee verkon liikennettä yhden edustajan osalta toimiakseen korrektisti ja yksimielisesti sen kanssa. Tämä vähentää sisäänpäintulevaa äänestysliikennettä jotka tulevat edustajilta tälle solmulle.

4) Kevyt: Kevyt solmu on myös luottava solmu joka vain tarkkailee liikeneettä tileille jolle se haluaa sallia minimaalisen verkon käytön.

5) Esilaskettu: Esilaskettu solmu palvelee tilikirjan kaikkia solmuja jotka tulevat yhteyteen (online). Tämä on tehty TCP yhteyden yli eikä UDP:n yli, koska siihen liittyy suuria määriä dataa joka vaatii kehittyneempää kontrollia.

B. Levykapasiteetti

Riippuen käyttäjän vaatimuksista, erilaiset solmukonfiguraatiot vaativat eri määrän tallennustilaa.

1) Historiallinen: Solmu joka on kiinnostunut pitämään täyttä historiallista dataa kaikista transaktioista vaatii maksimimäärän tallennustilaa.

2) Nykyinen: Kumuloidun taseen säilytyksen suunnittelusta johtuen, solmujen tarvitsee pitää kirjaa vain viimeisimmistä lohkoista jokaiselle tilille osallistuakseen konsensukseen. Jos solmu ei ole kiinnostunut pitämään kirjaa täydestä historiallisesta datasta, se voi pitää kirjaa vain viimeisistä lohkoista.

3) Kevyt: Kevyt solmu ei pidä lokaalia tilikirjadataa ja osallistuu vain verkkoon tarkkaillakseen aktivititeetteja niiltä tileiltä joiden kanssa se on mahdollisesti luomassa uusia transaktioita ja joiden salaisia avaimia se pitää sisällään.

C. CPU

1) Transaktioiden luominen: Solmu joka on kiinnostunut luomaan uuden transaktion pitää tuottaa Proof of Work nonce läpäistäkseen RaiBlocksin kiihdytysmekanismin. Eri laitteistojen laskentateho on suorituskykytestattu liitteessä A.

2) Edustaja: Edustajan tulee verifioida allekirjoitukset lohkoille, äänille ja myös tuottaa omat allekirjoitukset osallistuakseen konsensukseen. CPU resurssien määrä edustajasolmulle on huomattavasti pienempi kuin transaktioluonti ja sen pitäisi toimia millä tahansa nykyaikaisella tietokoneella.

3) Tarkkailija: Tarkkailijasolmu ei luo omia ääniään. Koska allekirjoitusluonti on minimaalista, CPU vaatimukset ovat melkein identtiset kuin edustajasolmun pyörittämiselle.

VIII. Yhteenveto

Tässä esityksessä esitimme viitekehyksen luottamattomalle, maksuttomalle, pienivasteaikaiselle kryptovaluutalle joka hyödyntää uudenlaista lohkosäleikkörakennetta ja delegoitua Proof of Stake äänestystä. Verkko vaatii minmaalisesti resursseja, ei korkeakulutusta louhintalaitteistoa, ja voi prosessoida suuren määrän transaktioita suoritusteholtaan. Kaikki tämä on saavutettu saamalla yksittäisiä lohkoketjuja jokaiselle tilille, eliminoimalla pääsyongelmat ja globaaleiden datarakenteiden tehottomuudet. Tunnistimme mahdolliset hyökkäysvektorit järjestelmään ja esitimme miksi RaiBlocks on näille hyökkäyksille resilientti.

Liite A
PoW laitteiston suorituskykytestit

Kuten aikaisemmin on mainittu, PoW on RaiBlocksin käytössä jotta verkon roskaa voidaan vähentää. Meidän solmuimplementaatio tuottaa kiihdytyksen joka voi hyödyntään OpenCL yhteensopivia GPUita. Taulukko I antaa oikean elämän suorituskykytestivertailun eri laitteistoille. Tällä hetkellä PoW kynnys on kiinteä, mutta adaptatiivinen kynnys on mahdollista implementoida kun keskimääräinen laskentateho kehittyy.

Taulukko I
Laitteiston PoW suorituskyky

Laite	Transaktiota sekunnissa
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

Huomionosoitus

Haluaisimme kiittää Brian Pughia tämän esityksen koostamisesta ja muotoilusta.

Viitteet

[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