Kasper Tontti

Lohkoketjupohjaiset hajautetut sovellukset

Tietotekniikan kandidaatintutkielma

21. joulukuuta 2018

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Kasper Tontti

Yhteystiedot: kajumito@student.jyu.fi

Työn nimi: Lohkoketjupohjaiset hajautetut sovellukset

Title in English: Blockchain based decentralized applications

Työ: Kandidaatintutkielma

Sivumäärä: 25+0

Tiivistelmä: Tämä tutkimus käsittelee lohkoketjuteknologiaan pohjautuvia hajautettuja sovelluksia. Tutkimuksessa käydään läpi lohkoketjun perusrakennetta ja älysopimuksia, jotka yhdessä muodostavat hajautetun sovelluksen. Hajautettuihin sovelluksiin tutkimuksessa tutustutaan eri arkkitehtuurimallien pohjalta, joka auttaa ymmärtämään eri tapoja hajautettujen sovelluksien suunnittelemiseen ja kehittämiseen.

Avainsanat: älysopimus, lohkoketju, hajautettu sovellus

Abstract: This thesis studies decentralized applications which bases on blockchain technology. Research goes through fundamental infrastructure of blockchain and smart contracts, which together construct a decentralized application. After fundamentals research will introduce you to basic architecture models of decentralized applications, which helps to understand different ways to design and develop these systems.

Keywords: smart contract, blockchain, decentralized application

Termiluettelo

Lohkoketju	Hajautettu tilikirja, johon tallennettu tieto on muuttumatonta
Älysopimus	Lohkoketjussa suoritettavaa ohjelmakoodia
Hajautettu sovellus	Ohjelma tai ohjelmisto, joka pohjautuu yhteen tai useampaan älysopimukseen
Vertaisverkko	Ilman keskitettyä kiinteää palvelinta toimiva verkko, jossa jokainen verkon jäsen toimii asiakkaana ja palvelimena
Transaktio	Tilitapahtuma, jossa arvoa siirretään tililtä toiselle
Konsensus	Yhteisymmärrys, jonka mukaan verkossa toimitaan
Solmu	Verkon toimija, joka on yhteydessä verkon muihin solmuihin linkin avulla
Kryptografia	Salaustekniikka, joka mahdollistaa turvallisen viestinnän verkossa

Kuviot

Sisältö

1 Johdanto

Älysopimuksien ja lohkoketjuteknologian avulla toisilleen ennalta tuntemattomat toimijat pystyvät siirtämään arvoa täysin autonomisesti. Älysopimus (engl. smart contract) on täysin koneellinen prosessi, mikä sijaitsee lohkoketjussa ja toimii täysin sopimukseen ohjelmoitujen sääntöjen mukaisesti (Buterin 2017). Tämä mahdollistaa sen, että jokainen osapuoli tietää sopimuksen säännöt ja toimintatavat etukäteen eikä luottamusta osapuolten välillä tarvita. Lohkoketjuun pohjautuvissa hajautetuissa sovelluksissa (engl. decentralized application) älysopimuksien avulla pystytään toimimaan turvallisesti ja edullisesti ilman kolmansia osapuolia ja välikäsiä, mikä mahdollistaa aivan uudenlaisten liiketoimintamallien ja prosessien syntymisen.

Lohkoketjupohjaiset hajautetut sovellukset ovat sovelluksia, jotka toimivat hajautetun vertaisverkon päällä ja pohjautuvat avoimeen lähdekoodiin (Prusty 2017). Lohkoketjuteknologiaan perustuvat hajautetut sovellukset muodostuvat yhdestä tai useammasta älysopimuksesta. Älysopimuksien tehtävänä hajautetuissa sovelluksissa on mahdollistaa käyttäjän ja lohkoketjun välinen vuorovaikutus tekemällä uusia tapahtumia eli transaktioita (engl. transaction) lohkoketjuun.

Lohkoketjuteknologiaa voidaan hyödyntää ja siitä voidaan puhua monessa eri kontekstissa monella eri tavalla. Tämän takia käsitteelle löytyy monia eri määritelmiä. Tässä tutkielmassa lohkoketjusta puhuttaessa lohkoketjun määritelmä on kryptografisesti suojattu vertaisverkossa (engl. peer-to-peer) toimiva hajautettu tilikirja, mihin voidaan lisätä ja päivittää tietoa yhteisen ymmärryksen eli konsensuksen (engl. consensus) avulla.

Tämä kandidaatintutkielma keskittyy tutkimaan, miten lohkoketjupohjaiset hajautetut sovellukset toimivat ja miten niitä suunnitellaan. Aluksi tutkielmassa esitellään, miten lohkoketjut toimivat ja mistä osista lohkoketjun lohkot muodostuvat. Kun lohkoketjun perustoiminta ja lohkon rakenne on selvä, keskitytään älysopimuksiin ja niiden toimintaan hajautetuissa sovelluksissa. Älysopimusalustoja tutkielmassa tarkastellaan julkisten (engl. public) ja yksityisten (engl. private) lohkoketjujen näkökulmista. Tarkastelussa apuna käytetään tunnetuimpia lohkoketjupohjaisia hajautettuja älysopimusalustoja, jotka tutkimuksen kirjoittamis-

vaiheessa ovat Ethereum ja Hyperledger Fabric. Tutkielman lopuksi esitellään, mistä komponenteista hajautetut sovellukset muodostuvat ja mitä tulisi ottaa huomioon, kun suunnitellaan hajautettuja sovelluksia. Tässä viimeisessä osiossa tutustutaan hajautettujen sovelluksien eri arkkitehtuurimalleihin kehittäjien kirjoittamien manuaalien ja alan ammattitaitoni perusteella.

Tutkielman tarkoituksena on antaa tekniikasta kiinnostuneille lukijoille peruskäsitys lohkoketjuteknologian toiminnasta, ja hyvä pohjatietämys hajautettujen sovelluksien toimintamalleista ja arkkitehtuurista. Tutkimuksessa käytetään tieteellisenä tutkimusmenetelmänä systemaattista kirjallisuuskatsausta, mutta tutkielma sisältää myös jonkin verran omaa pohdintaa ja synteettistä ajattelua. Tämä johtuu siitä, että lohkoketjuteknologia ja älysopimukset ovat verrattain uusi aihealue.

2 Lohkoketju

Lohkoketjuteknologia on saanut alkunsa digitaalisesta maksujärjestelmästä nimeltä Bitcoin, joka esiteltiin ensimmäistä kertaa lokakuussa vuonna 2008 valkopaperi julkaisussa “Bitcoin: A Peer-to-Peer Electronic Cash System” (Nakamoto 2008). Valkopaperi julkaisun taustalla oli salanimeä Satoshi Nakamoto käyttänyt entiteetti, jonka henkilöllisyyttä ei ole vieläkään todennettu. Julkisten lohkoketjujen asiantuntija Antonopoulos (2017) kuvailee Bitcoinin olevan yhdistelmä teknisiä innovaatioita, jotka muodostavat yhdessä vertaisverkkoon perustuvan hajautetun maksujärjestelmän. Bitcoinin kehittämisen jälkeen lohkoketjuteknologia on omaksuttu mullistavana teknologiana ja otettu laajempaan käyttöön. Lohkoketjuteknologian päälle on rakennettu useita muita hajautettuja alustoja ja sovelluksia, joista tässä tutkielmassa käsittelen pääasiassa kahta hajautettua älysopimusalustaa: Ethereum ja Hyperledger Fabric.

Nakamoton (2008) suunnitteleman täysin vertaisverkossa toimivan lohkoketjun kaikki solmukohdat (engl. nodes) ovat saman arvoisia, eli verkko on kokonaan hajautettu. Tämä tarkoittaa sitä, että suurimman osan verkon solmuista on toimittava yhteisymmärryksessä toistensa kanssa, jotta tiedetään, mikä on lohkoketjuun tallennetun datan yleismaailmallinen totuus ja tila. Lohkoketjuissa yhteisymmärrys eli konsensus saavutetaan eri konsensusmekanismien avulla, joista puhutaan enemmän tutkielman 2.2.1 kappaleessa.

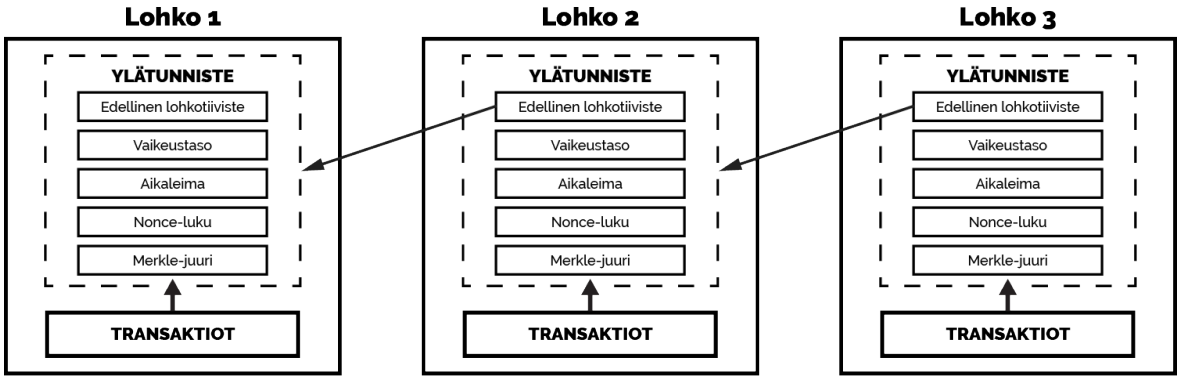
Vertaisverkossa verkon solmukohdat ylläpitävät omaa kopiota lohkoketjusta aina lohkoketjun alkulohkosta (engl. genesis block) lähtien. Solmukohdat päivittävät kopiota lohkoketjusta uusien lohkojen saapuessa varmentamalla ja linkittämällä lohkot lohkoketjuun viimeisimmän lohkon perään (Antonopoulos 2017). Hajautettu toimintamalli mahdollistaa sen, että verkkoon hyökkääminen ja lohkoketjun tietojen peukaloiminen on lähes mahdotonta. Verkon luotettavat solmukohdat pystyvät torjumaan hyökkäykset ja peukalointiyritykset, kun luotettavien solmujen määrä on yli puolet verkon solmuista (Nakamoto 2008). Koska vertaisverkoissa ei ole datan keskittymiä, yksittäisten solmujen kaatuessa koko järjestelmä ei kaadu, koska dataa pystytään edelleen jakamaan muista verkon solmukohdista.

Lohkoketju koostuu tietorakenteellisesti lohkoista, jotka on yhdistetty toisiinsa taaksepäin linkitettynä listana (Nakamoto 2008). Jokaiselle lohkoketjun lohkolle luodaan uniikki tiivis-

te (engl. hash) lohkon ylätunnisteesta (engl. header), jonka avulla lohkot pystytään tunnistamaan toisistaan (Antonopoulos 2017). Lohkot viittaavat aina edellisen lohkon ylätunnisteesta muodostuneeseen lohkotiivisteeseen (ks. kuvio 1) muodostaen näin taaksepäin linkitetyn listan eli lohkoketjun, mikä johtaa aina loppujen lopuksi lohkoketjun ensimmäiseen lohkoon eli alkulohkoon.

2.1 Lohkon rakenne

Lohko muodostuu kahdesta osasta: ylätunnisteesta, joka säilöö kaikki lohkon metatiedot, ja listasta kelvollisia transaktioita. Lohko on tietorakenteeltaan säiliö, jonka tehtävänä on säilöä kaikki todennetut transaktiot. Pääasiallisena tunnistimena lohkolle on vahvalla salaus algoritmilla lohkon ylätunnisteesta muodostettu kryptografinen tiiviste, joka on jokaiselle lohkolle uniikki. Lohko ei itsessään sisällä tätä tiivistettä esimerkiksi ylätunnisteessaan, vaan jokainen lohkoketjun solmukohta laskee itse lohkon tiivisteen vastaanottaessaan lohkon verkossa. (Antonopoulos 2017.)



Kuvio 1. Yleiskuva lohkoketjun perusrakenteesta

2.1.1 Ylätunniste

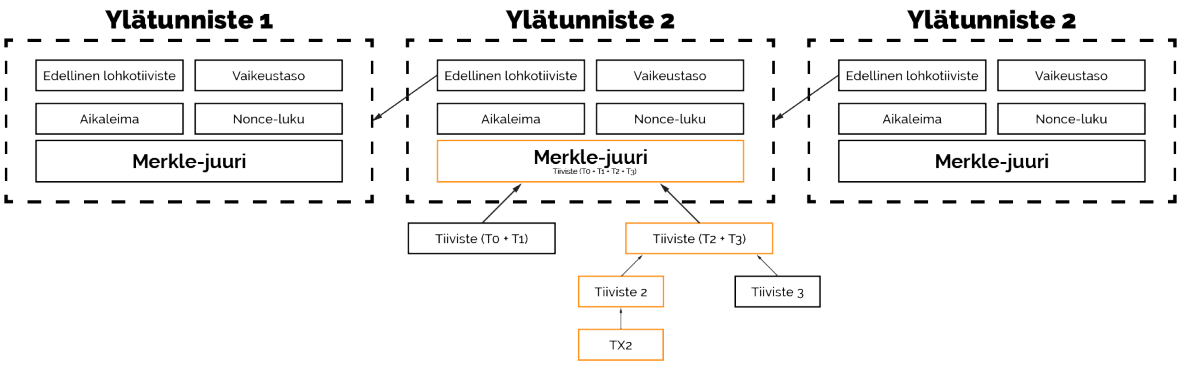
Lohkon ylätunniste sisältää lohkon metatiedot, ja se koostuu pääasiallisesti kolmesta eri osasta (Antonopoulos 2017). Ensimmäisenä lohkon ylätunnisteessa on viite edelliseen lohkotiivisteeseen, jonka avulla lohkot yhdistyvät toisiinsa. Toisena lohkon ylätunnisteessa on

louhintaan liittyvät metatiedot vaikeustaso (engl. difficulty), aikaleima (engl. timestamp), ja nonce-luku, joita käsitellään tarkemmin tutkielman kappaleessa 2.2.1. Kolmantena osana lohkon ylätunnisteessa on kaikista lohkon transaktioista muodostuvan Merkle-puun juuri tiivistettynä hajautusfunktion avulla.

Lohkon ylätunnisteessa sijaitsevan edellisen lohkon lohkotiivisteen viitteen ansiosta jokainen lohkoketjun lohko lukuun ottamatta alkulohkoa on riippuvainen aina edeltävästä lohkosta (Antonopoulos 2017). Tämä on yksi lohkoketjun keskeisimmistä elementeistä, sillä se takaa lohkoketjun muuttumattomuuden. Lohkoketjun lohkoihin tallennettuja transaktioita ei voida siis jälkeenpäin muuttaa ilman, että kaikkia muutettavaa lohkoa seuraavia lohkoja pakotetaan muuttumaan.

2.1.2 Merkle-juuri

Merkle ym. (1979) esittelivät tavan tallettaa dataa puu-tietorakenteeseen, jonka avulla suuren datamäärän paloiksi (engl. chunks) jakaminen mahdollisti pääsyn haluttuun osaan isoa tiedostoa käyttämällä huomattavasti vähemmän levytilaa. Tämä tapahtuu ’hajauta ja hallitse’ -tyylillä, jossa hajautetaan isoja paloja datasta pienen pieniin osiin, jonka jälkeen jokaisesta osasta muodostetaan rekursiivisesti pareittain tiiviste niin pitkään, että jäljelle jää vain yksi tiiviste, eli juuritiiviste. Tilan säästämiseksi lohkoketjun jokaisen lohkon kaikki transaktiot on tiivistetty hajautusfunktion avulla Merkle-puuhun, josta lohkon ylätunnisteeseen on tallennettu ainoastaan puun juuren tiiviste, eli Merkle-juuri (ks. kuvio 2) (Nakamoto 2008).



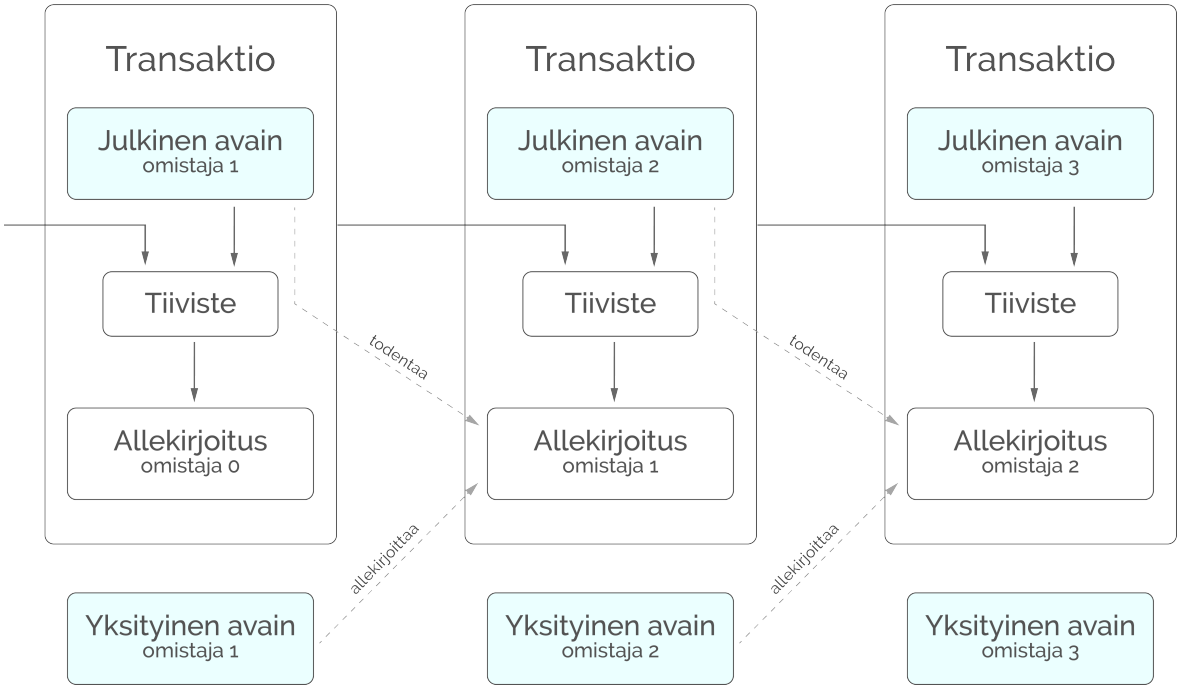
Kuvio 2. Merkle-juuren oksa transaktiolle TX2

Nakamoto (2008) hyödynsi tätä konseptia Bitcoinin kohdalla yksinkertaistetussa maksunvalvonnassa (engl. simplified payment verification, SPV). Merkle-juuren tiivistettä voidaan käyttää hyväksi transaktioista koostuvassa Merkle-puussa oikean reitin löytämiseksi haluttuun transaktioon. Merkle-juuri lohkon ylätunnisteessa mahdollistaa sen, että käyttäjän ei tarvitse ladata koko lohkoketjua vaan hän pystyy varmentamaan maksuja suorittamalla kyselyitä verkon solmukohtiin lataamalla pelkästään lohkojen ylätunnisteet. Merkle puun säästämän tilan ansiosta lohkoketjuteknologiaa voidaan hyödyntää kevyemmissä laitteissa, kuten esimerkiksi mobiililaitteissa ja erilaisissa IoT-komponenteissa.

2.1.3 Transaktiot

Koko lohkoketjuteknologia on rakennettu transaktioiden ympärille niin, että transaktioita pystytään luomaan, levittämään vertaisverkossa, todentamaan ja lisäämään tilikirjaan eli itse lohkoketjuun (Antonopoulos 2017). Transaktiolla tarkoitetaan kaikessa yksinkertaisuudessaan arvon siirtoa osoitteesta toiselle (Bashir 2017). Arvon siirron lisäksi esimerkiksi Ethereum lohkoketjussa transaktioiden avulla pystytään luomaan uusia älysopimuksia ja kutsumaan sopimuksien funktioita (Prusty 2017).

Arvon siirto lohkoketjussa tapahtuu digitaalisien allekirjoitusten avulla. Transaktioiden lähettämiseen ja vastaan ottamiseen tarvitaan tili, joka omistaa uniikin julkisen- ja yksityisen avaimen. Omistaja voi siirtää arvoa verkossa allekirjoittamalla edellisen transaktion tiivisteen ja seuraavan omistajan julkisen avaimen omalla yksityisellä avaimella (ks. kuvio 2) (Nakamoto 2008).



Kuvio 3. Transaktioiden allekirjoittaminen lohkoketjussa

2.2 Konsensus

Täysin hajautetussa vertaisverkossa, jossa jokainen solmu ylläpitää omaa kopiota koko lohkoketjusta, pitää olla tapa varmistaa jokaisen lohkoketjuun liitettävän lohkon oikeellisuus niin, että yksikään solmu ei pääse peukaloimaan lohkoketjuun tallennettua dataa (Prusty 2017). Solmujen tulee siis löytää yhteisymmärrys lohkoketjun tilasta ja sen sisältävästä datasta.

Baran (1964) esitteli artikkelissaan “On Distributed Communications Networks” ensimmäistä kertaa hajautetun verkkomallin konseptin. Täyden yhteisymmärryksen muodostaminen hajautetussa verkossa osoittautui kuitenkin tutkimuksessa suurimmaksi ongelmaksi. Lamport, Shostak ja Pease (1982) loivat tästä ongelmasta mielileikin, jonka avulla ongelmaa pystyttiin havainnollistamaan paremmin. Mielileikki tunnetaan nimellä Byzantin kenraalien ongelma (engl. The Byzantine Generals problem).

Byzantin kenraalien ongelman ratkaisemiseen kehitetyt algoritmit eivät kuitenkaan olleet tar-

peeksi nopeita ja kustannustehokkaita tuotantokäyttöön ennen, kuin Castro ja Liskov (1999) esittelivät “Practical Byzantine Fault Tolerance” -algoritminsa. Tämän tutkimuksen ja algoritmin pohjalta pystyttiin rakentamaan kehittyneempiä korkean suorituskyvyn algoritmeja ratkaisemaan Byzantin kenraalien ongelma. Ensimmäinen käyttökelpoinen verkon yhteisymmärrykseen eli konsensukseen muodostamiseen kehitetty konsensusmekanismi oli toteutettu Bitcoinissa ja se pohjautui proof-of-work algoritmiin (Bashir 2017).

2.2.1 Proof-of-work

Back (2002) esitteli teknisessä raportissaan “Hashcash - A Denial of Service Counter-Measure” toukokuussa 1997 suunnittelemansa internetin resurssien väärinkäyttöä kuten roskapostia torjuvan mekanismin nimeltä Hashcash. Mekanismi pohjautui Dworkin ja Naorin (1993) tekemään tutkimukseen “Pricing via Processing or Combatting Junk Mail”, jonka tarkoituksena oli vaatia käyttäjää suorittamaan kohtalainen määrä laskentaa päästäkseen käsiksi haluttuun resurssiin. Hashcash käytti laskennassa kustannustoimintona hyödyksi käyttäjän keskusyksikön prosessoria muodostamalla sillä tunnuksen, jolla pystyttiin todistamaan, että käyttäjä oli suorittanut tarpeeksi laskentaa suorittamaan kyseisen prosessin.

Nakamoto (2008) hyödynsi tehdyn työn todistusta (engl. proof-of-work) myös Bitcoinissa. Verkossa toimivat solmut koittavat muodostaa lohkon ylätunnisteesta mahdollisimman pienen tiivisteen SHA256-tiivistefunktion avulla. Laskentaa suoritetaan niin kauan, että joku verkon solmuista löytää kohdeluvun tai sitä pienemmän luvun. Jos solmun tuottama tiivistefunktio ei tuota tarpeeksi alhaista lukua, se muuttaa ylätunnisteessa sijaitsevaa nonce-lukua ja suorittaa tiivistefunktion ylätunnisteelle uudestaan. Tiivisteiden laskemisesta ja tarpeeksi pienen kohdeluvun löytämisestä muodostuvaa prosessia kutsutaan louhimiseksi (engl. mining).

Yhden verkon solmun löydettyä tarpeeksi pienen luvun, jossa on tarpeeksi monta nollaa tiivisteen alussa, voidaan varmistua siitä, että solmu on tehnyt tarvittavan määrän laskentaa tämän tiivisteen muodostamiseksi (Nakamoto 2008). Kun solmukohta on löytänyt tarpeeksi pienen tiivisteen, on sillä oikeus liittää uusi lohko lohkoketjuun ja ansaita siitä palkkio. Tämän jälkeen lohkoketjuun liitettyä lohkoa ei voida muuttaa ilman, että laskenta suoritet-

taisiin lohkolle ja sitä seuraaville lohkoille kokonaan uudestaan. Uuden lohkon luomista ja liittämistä lohkoketjuun yritetään pitää mahdollisimman tasaisena verkon vakauden ja tietoturvan takia (Buterin 2015). Tämä onnistuu säätelemällä lohkon ylätunnisteessa sijaitsevaa vaikeustasoa, joka tekee louhimisesta vaikeampaa, jos verkon laskentateho kasvaa ja lohkoja syntyy liian nopeasti tai helpompaa mikäli verkon laskentateho laskee ja lohkoja syntyy liian hitaasti.

Tehdyn työn todistus ratkaisee verkon päätösvallan määrittämisen ongelman. Suurinta osaa päätösvallasta edustaa pisin ketju, johon on investoitu suurin määrä laskentaa. Rehellisten solmujen hallitessa suurinta osaa verkon laskentatehosta ketju kasvaa nopeinten ja jättää jälkeensä kaikki muut kilpailevat ketjut. Hyökkääjän on täten lähes mahdotonta peukaloida lohkoja, koska hänen tulisi suorittaa laskenta uudelleen lohkolle ja sitä seuraaville lohkoille sekä saavuttaa ja ohittaa rehellisten solmujen tekemä työmäärä.

2.2.2 Proof-of-stake

Tehdyn työn todistuksen heikkoutena on laskennasta aiheutuvan energiankulutuksen määrä ja mahdollisen keskittyneisyyden riski. Tutkielman kirjoitushetkellä suurimman tehdyn työn todistusta hyödyntävän lohkoketjun Bitcoinin kokonaisenergiankulutus on noin 65TWh, mikä on verrattavissa pienen sisämaavaltion, kuten Tšekin energiankulutukseen (Digiconomist 2018). Tämä energiankulutus kasvaa koko ajan verkon kasvaessa ja on erittäin suuri haitta ympäristölle. Ongelmaa ollaan yritetty ratkaista tutkimalla vaihtoehtoisia tapoja muodostaa yhteisymmärrys täysin hajautetun verkon sisällä.

Varantotodistus (engl. proof-of-stake) on ympäristöystävällinen ja hajautetussa verkossa toimivaksi todettu konsensusmekanismi tehdyn työn todistuksen tilalle. Varantotodistus perustuu laskentatehon sijaan verkossa olevien todentajien sijoittamaan rahamäärään, jonka perusteella päätetään kuka saa luvan liittää seuraavan lohkon lohkoketjuun (Buterin 2016). Varantotodistus on tutkielman kirjoitushetkellä vielä kokeellisessa vaiheessa, ja sitä ollaan tällä hetkellä implementoimassa Casper -algoritmina Ethereum lohkoketjuun.

3 Älysopimus

Szabo (1996) keksi älysopimus-termin ja esitteli sen konseptin ensimmäistä kertaa artikkelissaan “Smart Contracts: Building Blocks for Digital Markets” kauan ennen lohkoketjuteknologian syntyä. Termi viittaa tietokoneohjelmaan, joka ylläpitää osapuolten välistä sopimusta ja suorittaa tarvittavia toimenpiteitä ohjelmakoodiin määriteltyjen ehtojen mukaisesti. Älysopimuksien ansiosta sopimuksen osapuolten välillä ei tarvitse luottamusta, koska kaikki toimenpiteet suoritetaan tietokoneohjelmassa, joka toimii poikkeuksetta sille annettujen ehtojen mukaisesti. Kun molemmat osapuolet luottavat ohjelmaan, luottamusta osapuolten välillä ei tarvita sopimuksen ehtojen toteutumiseksi.

Älysopimus-termille löytyy kirjallisuudesta monia erilaisia määritelmiä ja jaottelua, joten määrittelemme tässä tutkielmassa termin älysopimus tarkoittamaan yksinomaan ohjelmakoodia, joka on säilytetty, todennettu ja suoritettu lohkoketjussa (Alharby ja van Moorsel 2017). Lohkoketjuteknologian ja älysopimuksien avulla kaksi toisilleen täysin tuntematonta osapuolta voivat siirtää arvoa ilman luottamusta tai kolmansia osapuolia. Lohkoketjun tarjoaman muuttumattomuuden ansiosta voidaan todentaa sinne tallennettujen älysopimuksien toimivan täysin älysopimuksessa määriteltyjen ehtojen mukaisesti. Lohkoketjun hajautettu toimintamalli takaa myös sen, että älysopimus ja sen sisältämät toiminnot ovat saatavilla käyttäjälle koko ajan.

Bitcoin mahdollistaa yksinkertaisten älysopimuksien kehittämisen sille kehitetyn skriptikielen avulla (Bitcoin.org 2017). Skriptikieli Bitcoinille on toteutettu suoraviivaisesti ja ei ole Turing-täydellinen, jonka vuoksi se ei sovellu laajempien ja enemmän logiikkaa tarvitsevien älysopimuksien luomiseen. Tätä varten on kehitetty älysopimusalustoja, jotka mahdollistavat Turing-täydellisen tavan kehittää älysopimuksia lohkoketjuteknologian päälle.

3.1 Älysopimusalustat

Lohkoketjuteknologiaan pohjautuvan ja älysopimuksia käyttävän hajautetun sovelluksen kehittäminen on erittäin hankala ja aikaa vievä prosessi. Jokaisen lohkoketjuteknologiaa toteuttavan sovelluksen pitäisi pitää yllä omaa lohkoketjua, mikä tuottaa ongelmia etenkin pie-

nemmille sovelluksille mm. järjestelmän luotettavuuden, konsensuksen muodostamisen ja yhteensopivuuden kanssa (Buterin 2017). Tätä ongelmaa ollaan yritetty ratkaista luomalla älysopimusalustoja kuten Ethereum ja Hyperledger.

Älysopimusalustojen tarkoituksena on luoda perustavanlaatuinen pohjakerrosprotokolla hajautettujen sovellusten kehittämiseen. Yleensä älysopimusalustat sisältävät Turing-täydellisen ohjelmointikielen, joka mahdollistaa nopean ja turvallisen tavan kehittää älysopimuksia, jotka pystyvät olemaan vuorovaikutuksessa toistensa kanssa (Buterin 2017). Älysopimusalustojen ansiosta kuka tahansa voi kehittää älysopimuksia, joille voidaan määrittää omat ehdot ja toiminnot suunnittelematta koko lohkoketjua uudelleen.

Eri lohkoketjut ja älysopimusalustat voidaan kategorisoida niiden sisältämien ominaisuuksien perusteella. Yleisesti ottaen jaottelu tehdään sen perusteella kenellä on lupa liittyä ja olla osana verkkoa sekä ylläpitää jaettua tilikirjaa. Tässä tutkielmassa käytetään yleisintä jaottelua lohkoketjuille ja älysopimusalustoilla, mikä määräytyy verkon käytön luvanvaraisuuden perusteella julkisiin ja yksityisiin lohkoketjuihin.

3.1.1 Julkinen lohkoketju

Julkinen lohkoketju on kaikille avoin, sillä kuka vain voi liittyä ja osallistua tähän verkkoon. Tämä tarkoittaa sitä, että verkon data ja sen sisältämät toiminnot ovat kaikille näkyvissä ja käytettävissä. Julkisen lohkoketjun toteutukset rakentuvat yleensä jonkin tietyn tarkoituksen ympärille ja sisältävät käyttäjälle hyödyllisiä toimintoja, jotka kannustavat käyttäjiä liittymään verkkoon (Jayachandran 2017). Suosituimpia julkista lohkoketjua hyödyntäviä toteutuksia ovat tutkielmassa jo aikaisemmin esille tulleet digitaalinen maksujärjestelmä Bitcoin ja älysopimusalusta Ethereum.

3.1.2 Yksityinen lohkoketju

Yksityisessä eli luvanvaraisessa lohkoketjussa käyttäjä tarvitsee luvan lohkoketjun omistajalta tai verkon käyttäjiltä päästäkseen osallistumaan verkkoon. Yksityiset lohkoketjut sopivat parhaiten yrityksille ja yritysten väliseen toimintaan, jossa vaaditaan korkeamman tason tietosuojaa ja turvallisuutta. Yksityisessä lohkoketjussa ainoastaan transaktioon osallistuneet

tai sen tietoa tarvitsevat entiteetit ovat tietoisia transaktiosta ja pääsevät käsiksi siihen (Jayachandran 2017). Tällä hetkellä kaikista käytetyin viitekehys yksityisten lohkoketjujen kehittämiseen on The Linux Foundationin ylläpitämän Hyperledger-projektin osa-implementaatio Hyperledger Fabric.

4 Hajautettu sovellus

Täysin hajautetut vertaisverkkoihin perustuvat sovellukset saivat alkunsa 2000-luvun alkupuolella tiedostojen siirtämiseen rakennettujen protokollien pohjalta. Näistä tunnetuimpana Cohenin vuonna 2003 julkaisussaan “Incentives build robustness in BitTorrent” esittelemä BitTorrent-protokolla, joka on vieläkin käytössä. BitTorrent-protokollan avulla tiedoston lataajat samaan aikaan myös jakavat osia tiedostosta muille tiedoston lataajille. Tämän ansiosta BitTorrent-protokollaa käytettäessä tiedostojen siirtämiseen ei tarvita keskitettyjä palvelimia ylläpitämään ja jakamaan tiedostoja, koska tiedoston jakaminen on jaettu sen lataajien kesken ja on täten täysin hajautettua.

Laaja-alaisesti ajateltuna hajautetulla sovelluksella voidaan tarkoittaa siis sovellusta, joka perustuu täysin hajautettuun vertaisverkko infrastruktuuriin (Wood ja Antonopoulos 2018). Tässä tutkielmassa kuitenkin hajautetulla sovelluksella tarkoitetaan yhteen tai useampaan lohkoketjussa sijaitsevaan älysopimukseen pohjautuvaa sovellusta, jolle on luotu käyttöliittymä. Hajautetulle sovellukselle rakennettu käyttöliittymä mahdollistaa käyttäjän ja lohkoketjun välisen vuorovaikutuksen älysopimuksiin toteutettujen toimintojen pohjalta.

Wood ja Antonopoulos (2018) kuvailevat, että lohkoketjuteknologiaan pohjautuvien hajautettujen sovelluksien tarkoituksena on viedä World Wide Web kehityksessään seuraavalle tasolle. Uuden tason tarkoitus on lisätä web-sovellusten hajautuneisuutta vertaisverkkojen avulla ja täten parantaa toiminnan läpinäkyvyyttä sekä poistaa turhia välikäsiä ja kolmansia osapuolia. Tästä kehityksestä käytetään termiä Web3, joka on nimensä mukaisesti kolmas "versio"internetistä.

4.1 Hajautettu tilikirja

Lohkoketjuteknologian mahdollistaman kaikille verkon käyttäjille avoimen hajautetun tilikirjan avulla pystytään jakamaan ja pitämään kirjaa verkon transaktioista ilman yhdenkään keskitetyn tahon hallitsemista (Benos, Garratt ja Gurrola-Perez 2017). Tilikirjaan kirjatun tiedon eheys ja muuttumattomuus voidaan varmentaa kryptografisten tiivisteiden ja 2.2.1 kappaleessa esiteltyjen konsensusmekanismien avulla.

Ilman keskitettyä hallintoa ylläpitämän luottovapaan tietokantaratkaisun lisäksi ympäristön sisällä voidaan suorittaa ohjelmakoodia älysopimuksien avulla (Benos, Garratt ja Gurrola-Perez 2017). Tämä luo aivan uudenlaisien mahdollisuuden luoda sovelluksia, jotka ovat ytimeltään täysin luottovapaita.

4.2 Arkkitehtuuri

Hajautettujen sovelluksien suunnitteleminen ja kehittäminen on hankalaa, koska sovelluksen tulee olla joko täysin tai suurimmilta osin hajautettu. Sovelluksien kriittisimpien osien tulee sijaita ja toimia hajautetussa ympäristössä. Yleensä hajautettujen sovelluksien ei niin kriittiset osat, kuten esimerkiksi sovelluksen käyttöliittymä sijaitsevat keskitetyllä palvelimella.

Hajautettu sovellus koostuu yleensä kolmesta eri komponentista: älysopimuksista, käyttöliittymästä ja tietovarastosta. Seuraavaksi käymme lävitse eri malleja hajautetun sovelluksen kehittämiseen. Esimerkit pohjautuvat Ethereum -älysopimusalustaan, mutta mallit ovat myös sovellettavissa muihin olemassa oleviin älysopimusalustoihin.

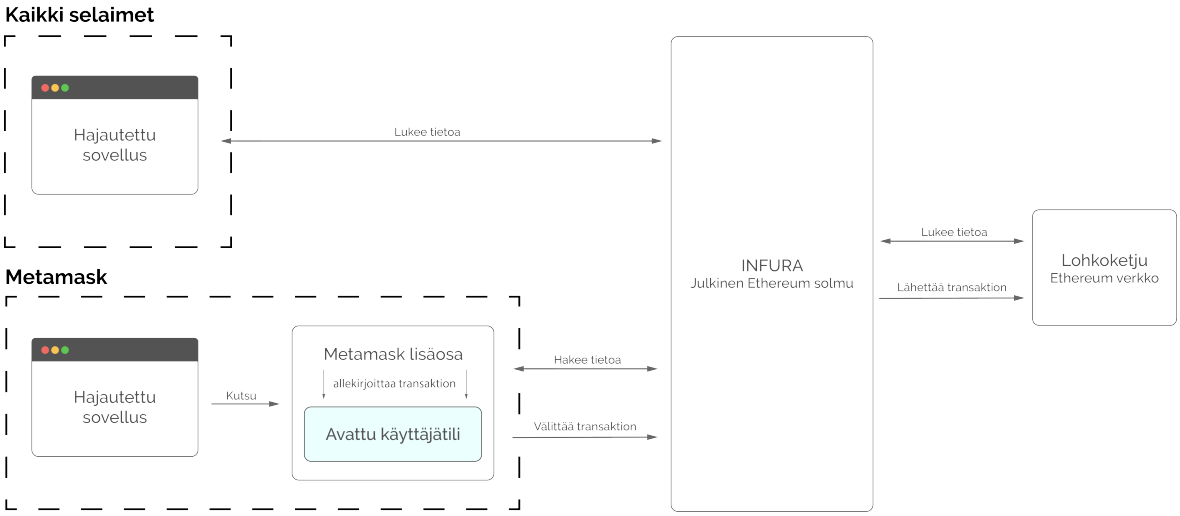
4.2.1 Asiakas-lohkoketju sovellukset

Asiakas-lohkoketju -arkkitehtuurimallissa tietoliikenne kulkee suoraan asiakkaan ja lohkoketjun välillä. Asiakkaan käyttämä käyttöliittymä yleensä sijaitsee keskitetyllä palvelimella, kuten esimerkiksi pilvipalvelun tarjoajalla ja sen tulee olla web3 yhteensopiva. Tämän lisäksi asiakkaan tulee olla yhteydessä Ethereum solmuun web3-palveluntarjoajan avulla, mikä mahdollistaa tiedon lukemisen ja kirjoittamisen lohkoketjuun. Yleisimmin yhdistäminen Ethereum solmuun tapahtuu Mist -selaimen tai Metamask -selainlisäosan avulla.

Mikäli asiakas ei käytä Mistiä tai Metamaskia hän voi yhdistää lohkoketjuun julkisen solmun kautta (ks. kuvio 4). Tämä mahdollistaa vain tiedon lukemisen lohkoketjusta, koska asiakkaalla ei ole henkilökohtaista avattua käyttäjätiliä. Ilman käyttäjätiliä asiakas ei voi allekirjoittaa ja lähettää transaktioita lohkoketjuun. Asiakkaan ei välttämättä tarvitse itse ylläpitää omaa julkista solmua vaan hän voi käyttää palveluntarjoajan, kuten Infuran tarjoamia julkisia solmuja ilmaiseksi.

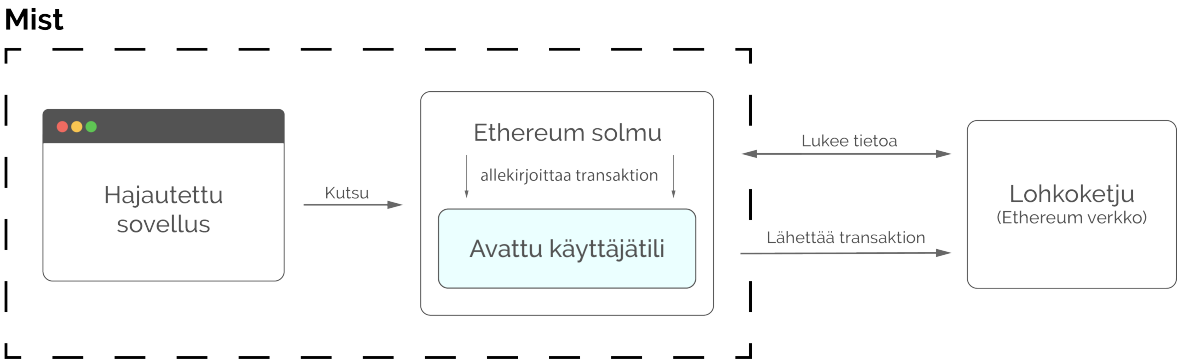
Metamask -selainlisäosa mahdollistaa kevyen ja helpon tavan tiedon lukemisen lisäksi myös lähettää transaktioita lohkoketjuun (ks. kuvio 4). Metamask tarjoaa asiakkaalle käyttäjätilin, jonka avulla hän voi allekirjoittaa transaktioita ja lähettää niitä julkiselle solmulle (Metamask 2015). Julkisia solmuja tarjoavaa Infuraa hyödynnetään myös Metamaskissa.

Huomioon tulee ottaa, että kolmannen osapuolen tarjoamia julkisia solmuja, kuten esimerkiksi Infuraa, käytettäessä asiakkaan tulee luottaa tähän kolmanteen osapuoleen. Asiakkaan tulee olla varma, että solmu, jota hän käyttää, myös tarjoaa oikeellista tietoa sekä toimittaa sille lähetetyt allekirjoitetut transaktiot lohkoketjuun.



Kuvio 4. Asiakas-lohkoketju -arkkitehtuurimalli julkisen solmun kautta

Ethereum Foundationin kehittämä ja ylläpitämä Mist -selain yhdistää lohkoketjuun suoraan sisäänrakennetun yksityisen Ethereum solmun kautta (ks. kuvio 5). Käyttäjätilin sisältävän yksityisen solmun avulla asiakas pystyy niin lukemaan tietoa lohkoketjusta, kuin allekirjoittaa ja lähettää transaktioita lohkoketjuun.

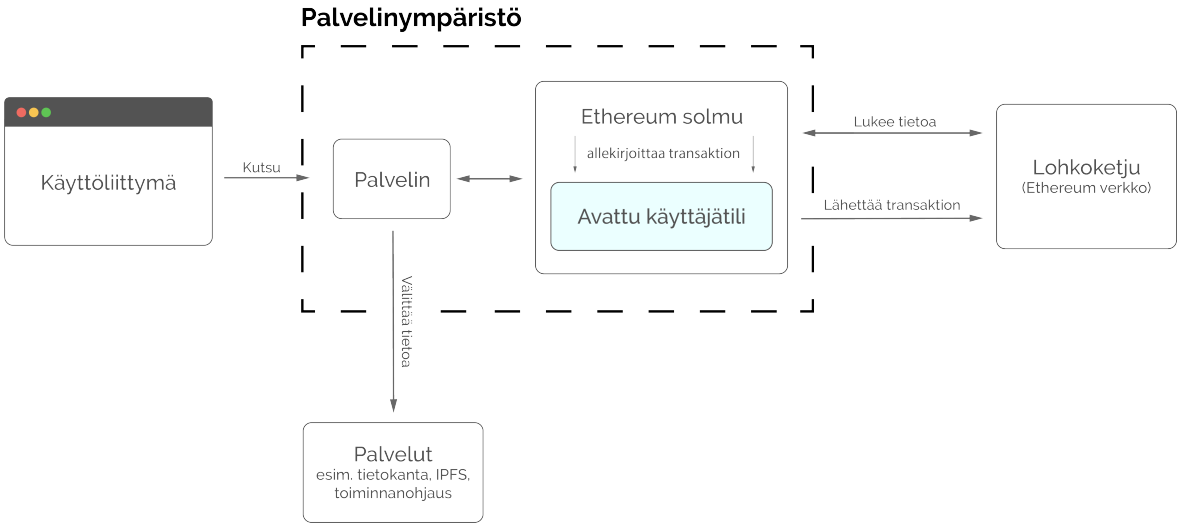


Kuvio 5. Asiakas-lohkoketju -arkkitehtuurimalli yksityisen solmun kautta

4.2.2 Palvelin-lohkoketju sovellukset

Palvelin-lohkoketju -arkkitehtuurimalli tuo järjestelmään palvelin -komponentin asiakkaan ja Ethereum verkon väliin (ks. kuvio 6). Tämä lisää järjestelmän keskittyneisyyttä, mutta tuo mukanaan uudenlaisia mahdollisuuksia. Palvelin mahdollistaa sovelluksen integroimisen kolmannen osapuolen palveluihin tai esimerkiksi tietokantaan.

Palvelimen avulla sovelluksesta tulee myös käyttäjäystävällisempi. Palvelin hoitaa asiakkaan puolesta käyttäjätililogiikan sekä tiedon vastaanottamisen ja transaktioiden lähettämisen. Tämän ansiosta asiakas voi yhdistää sovellukseen riippumatta web3 yhteensopivuudesta, koska palvelin hoitaa nämä asiat asiakkaan puolesta.



Kuvio 6. Palvelin-lohkoketju -arkkitehtuurimalli

5 Yhteenveto

Lohkoketju on vertaisverkossa toimiva hajautettu tilikirja, joka on muuttumaton. Kaikki verkon jäsenet voivat lukea ja kirjoittaa lohkoketjuun tietoa verkon yhteisen ymmärryksen eli konsensusmekanismin avulla. Lohkoketjun sisällä voidaan ajaa ohjelmakoodia sinne tallennettujen älysopimuksien avulla. Älysopimukset mahdollistavat hajautettujen sovelluksien kehittämisen, jotka toimivat ilman kolmansia osapuolia käyttäjältä käyttäjälle. Järjestelmän toimivuudesta voidaan olla varmoja, koska lohkoketjuun tallennetut älysopimukset ovat kaikille julkisia ja lohkoketju itsessään on täysin muuttumaton.

Teknologian epäkypsyyden takia hajautettujen sovelluksien kehittäminen on hankalaa. Suurimmat ongelmat täysin hajautettujen sovelluksien kehittämisessä tällä hetkellä ovat lohkoketjun skaalautuvuusongelmat sekä helppokäyttöisen ja sulavan käyttäjäkokemuksen luominen. Tämän hetken ratkaisuissa sovelluskohtaisesti joudutaan yleensä valitsemaan hyvän käytettävyyden ja sovelluksen hajautuneisuuden välillä. Suuren suosion ja kiihtyvän kehityksen ansiosta protokollia edistyneempien hajautettujen sovelluksien kehittämiseen luodaan jatkuvasti.

Eniten jatkotutkimustyötä tällä hetkellä tulisi tehdä julkisten lohkoketjujen skaalautuvuusongelmien ratkaisemiseksi. Tällä hetkellä esimerkiksi Ethereum lohkoketju pystyy prosessoimaan vain 15 transaktiota sekunnissa. Hajautettujen sovelluksien yleistyessä transaktionopeus tulee pullonkaulaksi monelle julkista lohkoketjua hyödyntävälle sovellukselle. Skaalautuvuusongelmiin on yritetty etsiä ratkaisua Ethereum verkossa ’Sharding’ -tekniikan ja ’Layer 2’ -tekniikoiden, kuten State Channels ja Plasma avulla.

Lähteet

Alharby, M., ja A. van Moorsel. 2017. “Blockchain-based Smart Contracts: A Systematic Mapping Study”. *ArXiv e-prints* (lokakuu). arXiv: 1710.06372 [cs.CR].

Antonopoulos, Andreas M. 2017. *Mastering Bitcoin: Unlocking Digital Crypto-Currencies.* 2nd. O’Reilly Media, Inc. ISBN: 1449374042.

Back, Adam. 2002. “Hashcash - A Denial of Service Counter-Measure”.

Baran, P. 1964. “On Distributed Communications Networks”. *IEEE Transactions on Communications Systems* 12, numero 1 (maaliskuu): 1–9. ISSN: 0096-1965. doi:10.1109/TCOM.1964.1088883.

Bashir, Imran. 2017. *Mastering Blockchain.* Birmingham, Packt Publishing.

Benos, Evangelos, Rod Garratt ja Pedro Gurrola-Perez. 2017. “The economics of distributed ledger technology for securities settlement”.

Bitcoin.org. 2017. “Bitcoin Developer Guide”. https://bitcoin.org/en/developer-guide.

Buterin, Vitalik. 2015. *On Slow and Fast Block Times.* Blog. https://blog.ethereum.org/2015/09/14/on-slow-and-fast-block-times/.

———. 2016. *Proof of Stake FAQ.* wiki. https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQ.

———. 2017. “A Next-Generation Smart Contract and Decentralized Application Platform”. *White-Paper.* https://github.com/ethereum/wiki/wiki/White-Paper.

Castro, Miguel, ja Barbara Liskov. 1999. “Practical Byzantine Fault Tolerance”. Teoksessa *Proceedings of the Third Symposium on Operating Systems Design and Implementation,* 173–186. OSDI ’99. New Orleans, Louisiana, USA: USENIX Association. ISBN: 1-880446-39-1. http://dl.acm.org/citation.cfm?id=296806.296824.

Cohen, Bram. 2003. “Incentives build robustness in BitTorrent”. Teoksessa *Workshop on Economics of Peer-to-Peer systems,* 6:68–72.

Digiconomist. 2018. “Bitcoin Energy Consumption Index”. Viitattu 23. huhtikuuta 2018. https://digiconomist.net/bitcoin-energy-consumption.

Dwork, Cynthia, ja Moni Naor. 1993. “Pricing via Processing or Combatting Junk Mail”. Teoksessa *Advances in Cryptology — CRYPTO’ 92,* toimittanut Ernest F. Brickell, 139–147. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-540-48071-6.

Jayachandran, Praveen. 2017. “The difference between public and private blockchain”. https://www.ibm.com/blogs/blockchain/2017/05/the-difference-between-public-and-private-blockchain/.

Lamport, Leslie, Robert Shostak ja Marshall Pease. 1982. “The Byzantine Generals Problem”. *ACM Trans. Program. Lang. Syst.* (New York, NY, USA) 4, numero 3 (heinäkuu): 382–401. ISSN: 0164-0925. doi:10.1145/357172.357176. http://doi.acm.org/10.1145/357172.357176.

Merkle, Ralph Charles, Ralph Charles erkle, Ralph Charles Yerkle, Ate Students, Steve Pohlig, Raynold Kahn ja Dov Andleman. 1979. *Secrecy, authentication, and public key systems.* Tekninen raportti.

Metamask. 2015. “Metamask extension”. https://github.com/MetaMask/metamask-extension.

Nakamoto, Satoshi. 2008. “Bitcoin: A Peer-to-Peer Electronic Cash System”. *White-Paper.* https://bitcoin.org/bitcoin.pdf.

Prusty, N. 2017. *Building Blockchain Projects.* Packt Publishing, Limited. ISBN: 9781787122147. https://books.google.fi/books?id=oq1EvgAACAAJ.

Szabo, Nick. 1996. “Smart Contracts: Building Blocks for Digital Markets”. https://web.archive.org/web/20160703083010/http://szabo.best.vwh.net/smart_contracts_2.html.

Wood, Gavin, ja Andreas M. Antonopoulos. 2018. *Mastering Ethereum.* Early Release. O’Reilly Media, Inc. ISBN: 9781491971949.