Bitcoin: Kasutajalt-kasutajale elektrooniline raha süsteem

Satoshi Nakamoto
https://bitcoin.org/en/bitcoin-paper

Eesti keele tõlge:
bitcoin@liigalihtne.ee
@LiigalihtneBTC, liigalihtne.ee/bitcoin

Abstrakt. Otseselt kasutajalt-kasutajale vahetatav elektrooniline raha võimaldaks veebimakseid saata ühelt poolelt teisele ilma ühtegi finantsasutust läbimata. Digiallkirjad lahendaksid osa probleemist, kuid nõuavad ikkagi kolmanda osapoole olemasolu, kes kinnitaks, et kontoseisud on õiged ja et ei toimuks topelt kulutusi. Me pakume lahendust kasutajalt-kasutjale võrgu topeltkulutamise probleemile. Võrk aja-märgistab tehingud räsides(hash -eng, tõlk.) need jooksvasse ahelasse räsimisel põhineva töö tõendite jadana, tekitades registri mida ei ole võimalik muuta ilma eelnevalt tehtud tööd uuesti tegemata. Kõige pikem ahel ei teeni ainult järjestuse tõestamise eesmärki vaid ka tõestust selle kohta, et see tuli suurimast protsessori võimsusega fondist. Seni kaua, kui suurem osa protsessori võimsusest on heatahtlike serverite käes, genereerivad nad pikima ahela ja liiguvad kiiremini ründajatelt eest. Võrk ise vajab minimaalselt struktuuri. Ülekandeid/sõnumeid saadetakse võrgus heasoovlikkuse alusel ning serverid saavad lahkuda ja ühineda võrguga igal ajal, aksepteerides ühinedes pikimat töötõenditel (Proof Of Work -tõlg.) põhinevat ahelat, mis loodi sellel ajal, kui nad ei olnud võrguga ühenduses.

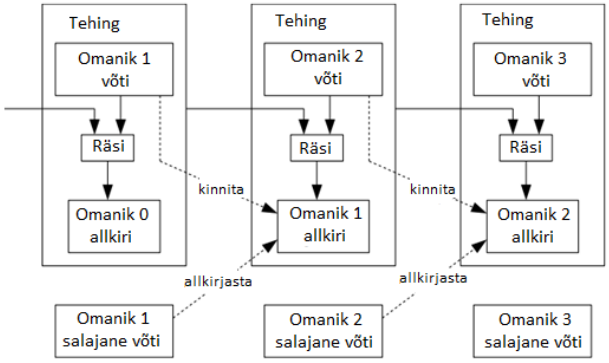
1. Sissejuhatus

Internetikaubandus on täielikult sõltuvuses finantsasutustest, kui usaldusväärsetest kolmandatest pooltest e-makstete teostamisel. Kuigi süsteem töötab piisavalt hästi suurema osa maksete tegemisel, kannatab see siiski usaldusel põhineva mudeli loomupäraste nõrkuste all. Täielikult tagasipöördumatud tehingud ei ole reaalselt võimalikud kuna finantsasutusel puudub võimalus vältida maksetega seotud vaidlusi. Vahendamine aga tõstab tehingu tasusid, seades piire praktilise tehingu summa suurusele ja lõikab välja võime teostada igapäevaseid väikseid makseid. Samuti on kahju suurem sellest, et süsteem ei võimalda luua tagasipöördumatuid makseid tagastamatute teenuste juures. Tagasipööratavate maksetega kaasneb suurem vajadus usalduse järgi. Kaupmees peab olema teadlikum oma kliendist, mis toob talle rohkem infot kliendist, kui tehingu tegemise jaoks vajalik oleks. Kindel protsent pettustest on aksepteeritud kui vältimatud. Neid kulusid ja ebakindlusi saab küll vältida makstes isiklikult sularaha kasutades, kuid olemas ei ole ühtegi süsteemi, mis võimaldaks teha tehinguid üle sidevõrgu, ilma kolmandaid osapooli usaldamata.

Usaldusväärse kolmanda pooleta läheb vaja elektroonilist krüptograafilistel tõenditel põhinevat maksete süsteemi, mis võimaldaks mõlemal osapoolel suhelda ja teha tehinguid otse "kasutajalt-kasutajale" ilma kolmandaid osapooli vajamata. Ülekanded, mis on arvutuslikult ebapraktilised, et tagasi pöörata, kaitseksid müüjaid pettusest ja ostjate kaitseks saab hõlpsasti rakendada tavapäraseid deposiidimehhanisme. Sellel paberil, me esitleme lahendust topelt kulutuste probleemile kasutades kasutajalt-kasutajale jagatud ajatemplite serverit, et genereerida arvutuslikku tõendit ülekannete ajalise järjestuse kohta. Süsteem on turvaline seni kuni ausad võrguosalised kollektiivselt kontrollivad rohkem arvutusvõimsust, kui pahatahtlik osalejate grupp.

2. Ülekanded

Me defineerime elektroonilse mündi, kui digitaalsete allkirjade ahela. Iga uus omanik saadab mündi edasi digitaalselt allkirjastades eelneva ülekande räsi ja järgmise omaniku avaliku võtme ning lisades need mündi allkirjade ahela lõppu. Makse saaja saab kontrollida alkirju, et kinnitada omanike ahelat.

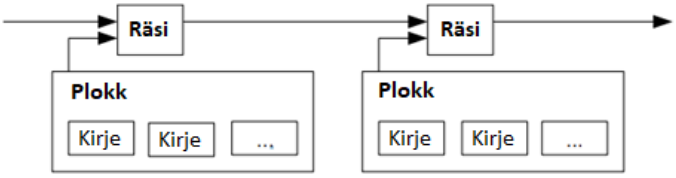


Probleem on aga see, et makse saaja ei saa kinnitada, et üks mündi omanikest seda kaks korda ei kulutanud. Tavapärane lahendus oleks lisada usaldatav kolmas pool, kes kontrollib igat makset, et ei esineks topelt kulutamisi. Peale igat makset peaks münt minema tagasi mündi väljastaja juurde, kes väljastab uue mündi ja ainult müntide puhul, mis on tulud otse väljasta juurest, saab kindel olla, et neid ei ole kasutatud topelt maksete tegemiseks. Selle süsteemi suurimaks miinuseks on see, et kogu usaldus süsteemi toimimisse langeb asutusele, kes münte väljastab, sest iga tehing peab nende juurest läbi käima nagu pangas.

Meil on vaja viisi kus makse saaja saab kindel olla, et eelmine omanik ei allkirjastanud teisi makseid kasutades seda sama digitaalset münti. Meie lahendus on ainult aksepteerida kõige varasemat mündiga tehtud makset, et me ei peaks mõtlema hilisematele katsetele sama omaniku poolt münti veel kord kasutada. Ainus viis veendumaks tehingute puudumises on olla teadlik kõigist maksetest. Keskse rahapaja põhises mudelis oli väljastaja teadlik kõigist maksetest ja otsustas milline makse esimesena temani jõudis. Selleks, et seda saavutada ilma kolmanda osapooleta, peavad ülekanded olema avalikult esitatud[1] ja meil peab olema võrguosaliste süsteem, kus kõik osalised tunnistavad ühte ja sama maksete toimumise järjekorda. Makse saaja vajab tõestust, et iga makse tegemise hetkel oli see osalejate poolt kinnitatud, kui kõige esimene makse.

3. Ajatemplite server

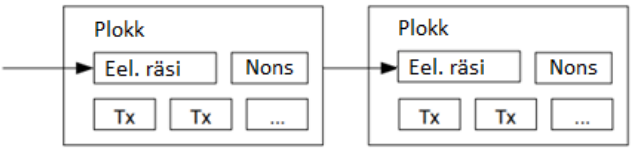
Lahendus, mida pakume algab ajatemplite serveriga. Ajatemplite server võtab ajatempleid vajavate tehingute ploki räsi ja jagab seda avalikult nagu ajalehte või foorumipostitust. Ajatempel tõendab, et andmed eksisteerisid kindlal ajal kuna need kaasati avaldatud räsisse. Iga ajatempel sisaldab ka eelmist ajatemplit oma räsis, tekitades ahela, kus iga uus ajatempel tugevdab sellele eelnevaid.



4. Töö tõendamine (Proof-of-Work)

Luues kasutajalt-kasutajale põhimõttel ajatemplite serveri on meil vaja töö tõendamise süsteemi, sarnaselt Adam Backi *Hashcash*-le[6], mitte nagu ajaleht või foorum. Töö tõendamine hõlmab räsitud väärtuse skannimist. Näiteks SHA-256 puhul algab räsi nullbitiga. Keskmine nõutav töö nullbitiga algavate räside leidmiseks on eksponensiaalne ja kontrollitav ühe räsi käivitamisega.

Meie ajatemplite võrgule loome me töö tõendamise süsteemi, lisades nons -e plokki seni, kuni oleme leidnud väärtuse mis annab ploki räsile vajaliku arvu nulle. Kui protsessor on piisavalt tööd teinud rahuldamaks töö-tõendamise algoritmi, ei saa plokki enam muuta, ploki loomiseks tehtud tööd uuesti tegemata. Kuna järgnevad plokid on aheldatud selle ploki külge tuleb selle ploki muutmiseks teha ka uuesti töö, mis kulus vahepeal järgnevate plokkide loomiseks.



Töö tõendamine lahendab ka küsimuse osalejate esindatuse osas suurte otsuste tegemisel. Kui hääletamine oleks põhimõttel: üks ip - üks hääl, oleks seda lihtne ära kasutada kellelgi, kellel on ligipääs paljudele ip -dele. *Proof of work* algoritmi juures on see sisuliselt: üks protsessor - üks hääl. Enamuse otsus esitletakse pikima plokiahelaga, mille loomiseks on kõige rohkem arvutusvõimsust panustatud. Kui suurem osa arvutusvõimsusest on kontrollitud ausate osalejate poolt, kasvab õige ahel kiiremini, kui mõni võistelda prooviv ahel. varasema ploki muutmiseks peab ründaja tegema uuesti selle ploki ja kõigile sellele järgnevatele plokkidele kulutatud töö ja järgi jõudma ausate osalejate loodud ahelale ja sellest ette jõudma. Hiljem näitame, kuidas aeglasema ründaja võimalused, ausalt loodud ahelale järgi jõuda kaovad eksponensiaalselt iga järgneva ploki loomisega.

Et kompenseerida riistavara kiiruse kasvu ja varieeruvat huvi plokkide loomisel, sõltub töö tõendamise raskusaste ühes tunnis loodavate plokkide arvu jooksvast keskmisest. Kui plokke luuakse liiga kiiresti, siis tõstetakse raskuaste kõrgemaks.

5. Võrk

Võrgu tööshoidmiseks on vajalik järgnev:

1) Uued tehingud edastatakse kõigile serveritele.
2) Iga server kogub uued tehingud plokki.
3) Iga server töötab leidmaks keerulist töötõendit oma plokile.
4) Leides töötõendi, edastab ta ploki kõigile teistele serveritele.
5) Serverid aksepteerivad ploki ainult siis, kui kõik selles sisalduvad ülekanded on õiged ja ei ole juba varasemalt kulutatud.
6) Osapooled väljendavad ploki aksepteerimist luues ahelasse järgmise ploki, mis sisaldab eelneva, aksepteeritud ploki räsi.

Serverid aksepteerivad alati pikimat ahelat, kui õiget ja töötavad selle kallal, et seda kasvatada. Kui kaks serverit edastavad samaaegselt kaks erinevat plokki võrgule, võivad mõned osapooled vastu võtta ühe ploki ja teised teise. Sellisel juhul töötavad nad esimesena saabunud ploki kallal, kuid säilitavad ka teise haru ahelast juhuks, kui see teine kasvab pikemaks. Üks ahelatest lõppeb, kui uus plokk edastatakse võrgule ja teine ahel kasvab pikemaks. Serverid, mis töötasid lühema ahela kallal lülituvad nüüd ümber pikema ahela juurde.

Uued tehingud ei pea ilmtingimata kõigi osapoolteni jõudma seni, kuni need jõuavad paljude serveriteni, kes lisavad need plokki. Samuti on plokiedastuses lubatud katkenud ühendused. Kui serverini ei jõua plokki, küsib ta seda võrgust järgmise ploki saabumisel.

6. Stiimul

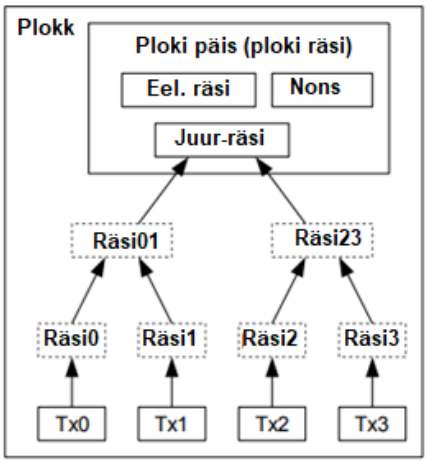
Kokkuleppe järgi on esimene tehing plokis eriline kuna see tekitab juurde uue mündi, mis läheb ploki looja omandusse. See annab stiimuli serveritele võrku toetada ja samas loob viisi kuidas münte ringlusesse lasta ilma keskse münte väljastava asutuseta. Uute müntide konstantse koguse pidev lisamine on analoogne kulla kaevurite kulutustega ressurssidele kulla ringlusse lisamiseks. Meie puhul kulutatakse protsessorite tööaega ja elektrit.

Stiimuliks võivad olla ka ülekandetasud. Ülekande väljundväärtus on väiksem, kui sisendi väärtus ja see väärtuste vahe ongi ülekandetasu, mis lisatakse ploki loomise tasule. Kui ettemääratud kogus münte on ringlusesse jõudnud, jääbki tasuks plokkide loomise eest ainult ülekandetasud ja süsteem muutub täielikult inflatsioonivabaks.

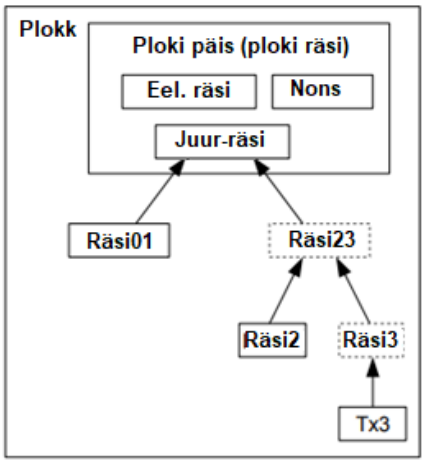
Stiimul võib motiveerida ka kaevandajaid jääma ausaks. Kui ahne ründaja suudab kokku panna rohkem arvutusvõimsust, kui ausad osalised siis tal on valida, kas petta inimesi pöörates oma maksed tagasi ja varastada tagasi oma kulutatud mündid või jääda ausaks. Ta võib avastatda, et reeglite järgi mängides ja koguaeg uusi plokke luues võib ta teenida rohkem uusi münte kui teised asjaosalised ja siis pole põhjust süsteemi ning oma varanduse kindlust õõnestada.

7. Kettaruumi taaskasutamine

Ajaks mil viimane tehing mündis on mattunud piisava hulga plokkide alla, võib selle välja heita, säästmaks kõvaketta ruumi. Et seda teha ilma ploki räsi lõhkumata, on tehingud räsitud “räsi puusse“ (*Merkle tree* -tõlg.) [7][2][5], kus on räsitud ainult plokkide juured. Seejärel saab teha plokid kompaktsemateks, lüües harusid vähemaks. Sisemisi räsisid ei ole tarvis säilitada.



Tehingud räsitud Merkle puusse

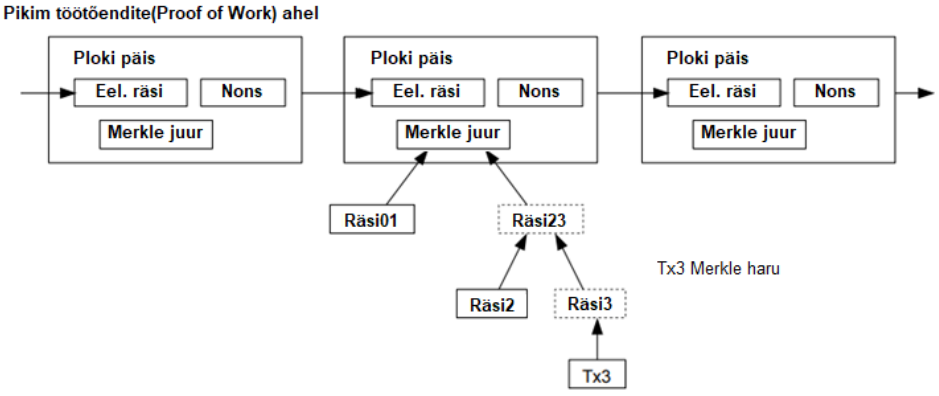


Peale Tx0-2 välja lõikamist plokist

Ploki päise maht ilma ühegi ülekandeta jääb 80 baidi ringi. Eeldusel, et plokke luuakse iga 10 minuti järel, saame arvutuse: 80 baiti * 6 *24*365=4,2MB aastas. Kui 2008 aastal on arvutisüsteemil tavaliselt 2GB vahemälu ja Moore -i seadus(Moore's Law -tõlg.) ennustab praeguseks mälumahu kasvuks 1,2GB aastas siis ei tohiks andmete hoiustamine probleemiks olla, isegi kui plokipäiseid tuleb hoiustada mälus.

8. Lihtsustatud maksete kinnitamine

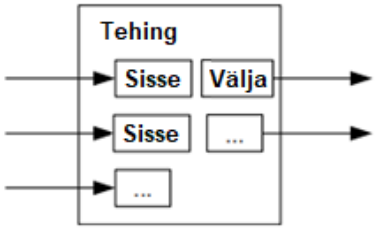
Maksete tõesust on võimalik kontrollida ka ilma täiemahulist serverit jooksutamata. Kasutajal tuleb alles hoida koopia kõige pikema töökindla ahela ploki päistest, mille ta saab serveritelt päringuid tehes, kuni on veendunud, et tal on kõige pikem ahel ja hankida Merkle puu haru, mis ühendab ülekannet plokiga, kus sellele on lisatud ajatempel. Ta ei saa küll ise ülekannet vaadata aga sidudes selle kindla kohaga plokiahelas, näeb ta, et võrk on selle aksepteerinud ja hiljem lisatud plokid kinnitavad veelgi, et võrk on tehingu õigeks lugenud.



Seni kuni võrku kontrollivad ausad serverid, on selline kontrollimine usaldusväärne, kuid kui suurimat võimsust kontrollib pahatahtlik ründaja, muudab see võrgu haavatavamaks. Ehkki serverid saavad tehinguid ise kontrollida, võib ründaja, kes valitseb võrgus enamusvõimsust, lihtsustatud meetodi üle kavaldada tootes fabritseeritud ülekandeid.

9. Väärtuse jagamine ja kombineerimine

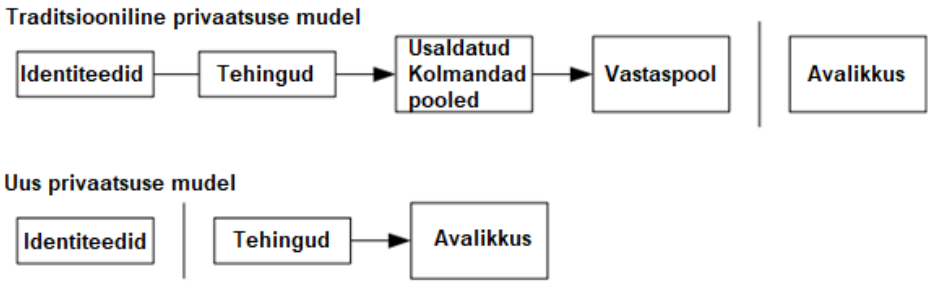
Olgugi, et münte oleks võimalik käsitleda eraldi, oleks kohmakas teha uus ülekanne iga sendi ülekandmise jaoks. Et ülekandeid oleks võimalik jagada ja uuesti kombineerida, sisaldavad nad mitmeid sisendeid ja väljundeid. Tavaliselt on seal üks sisend varasemast suuremast ülekandest või siis mitu sisendit väiksemates summades. Lisaks veel vähemalt kaks väljundit: üks makse jaoks ja üks tagastamaks vahetusraha saatjale, kui seda peaks tekkima.



Tuleb märkida ka, et väljavõte, kus tehing sõltub mitmest ülekandest ja need omakorda sõltuvad veel mitmest, ei ole siin probleem. Tehingute ajaloost ei ole kunagi vajadust täpset eraldiseisvat koopiat eraldada.

10. Privaatsus

Traditsiooniline pangandusmudel suudab tagada privaatsust piirates juurdepääsu andmetele mis sisaldavad infot tehingupoolte ja kolmanda poole kohta. Vajadus kõiki tehinguid avalikult edastada välistab siin selle meetodi. Küll aga saab privaatsust siiski säilitada, rikkudes infovoogu teises kohas: hoides avalikke võtmeid anonüümsetena. Avalikkus küll näeb, et keegi saadab mingi summa kellelegi teisele, kuid puudub info sidumaks tehingut konkreetse isikuga. See on sarnane infole, mis on aktsiaturus avalikkusele edastatud, kus aeg ja tehingu suurus tehakse avalikuks ütlemata, kes tehingu osapoolteks on.



Lisa tulemüürina võiks kasutada uut võtmepaari iga tehingu jaoks, et poleks võimalust siduda neid ühe omanikuga. Mitme sisendiliste ülekannete puhul, kus tuleb avalikustada, et kõik sisendid kuulusid samale omanikule, on mõningane sidumine siiski vältimatu.

11. Arvutused

Oletame stsenaariumi, kus ründaja üritab luua alternatiivset plokiahelat kiiremini, kui ausad võrguosalised. Isegi, kui ta seda suudab siis ei ava see võrku meelevaldsetele muutustele nagu raha õhust loomine või teisele omanikule kuuluvate müntide omastamist. Serverid ei kiida heaks kehtetuid ülekandeid makseteks ja ausad võrguosalised ei aksepteeri ühtegi plokki, mis neid ülekandeid sisaldavad. Ründaja saab muuta ainult oma ülekandeid, et sealt raha tagasi võtta.

Võiduajamist ausa ahela ja ründaja ahela vahel võib iseloomustada, kui binomiaalset juhuslikku jalutuskäiku(Binomial Random Walk - tõlg.). Edukas tulemus on, kui aus ahel muutub pikemaks ühe ploki võrra, viies edumaa +1 -ni ja mitte-edukas tulemus on see, kui ründaja ahel saab pikemaks ühe ploki võrra vähendades vahemaad -1 võrra.

Tõenäosus, et ründaja jõuab defitsiidist järgi on analoogne “mänguri hävingu(Gambler’s Ruin - tõlg.)” probleemile. Oletame, et lõputu krediidiga mängur alustab kaotusest ja mängib lõputu jada käsi, et jõuda nulli tagasi. Me saame välja arvutada tõenäosuse, kas mängur jõuab nulli või ründaja ausale ahelale järgi[8]:

p=tõenäosus, et aus server leiab järgmise ploki
q=tõenäosus, et ründaja leiab järgmise ploki
q_z=tõenäosus, et ründaja jõuab kunagi järgi olles z arvu plokke maas

$$q_z = \begin{cases} 1 & \text{if } p \le q \\ (q/p)^z & \text{if } p > q \end{cases}$$

Kasutades meie oletust, et p > q, siis tõenäosus, et ründaja järgi jõuab, kukub eksponensiaalselt kuna plokkide arv millele ründaja peab järgi jõudma suureneb. Kuna panused on tema vastu siis ta šansid muutuvad kaduvväikseks, kui tal just ei ole õnne rünnaku alguses kohe ette saada.

Nüüd kaalume, kui kaua uue tehingu saaja peab ootama, enne kui oleme piisavalt kindlad, et saatja ei saa tehingut muuta. Eeldame, et saatja on ründaja, kes soovib panna saaja uskuma, et ta maksis talle mingi aeg ja siis mõne aja möödudes üritab ülekande tagasi pöörata endale. Saajat küll teavitatakse sellest, kui see juhtub, kuid saatja loodab, et on juba liiga hilja.

Makse vastuvõtja loob uue võtmete paari ja annab avaliku võtme saatjale vahetult enne allkirjastamist. See hoiab ära saatja võimaluse valmistada ette piisavalt pikk ahel, et hea õnne korral õigest ahelast ette jõuda ja siis sel hetkel tehing sooritada. Kohe, kui ülekanne on sooritatud hakkab ebaaus saatja töötama salaja teise paralleelse ahela kallal, mis sisaldab alternatiivset versiooni tema tehingust.

Saaja ootab kuni tehing on lisatud plokki ja z arv plokke on juurde lisatud. Ta ei tea ründaja tehtud edusammude täpset suurust, kuid eeldades, et ausad plokid võtsid ploki kohta keskmiselt eeldatava aja, on ründaja potentsiaalne edasiminek eeldatava väärtusega Poissoni jaotus:

$$\lambda = z\frac{q}{p}$$

Et saada tõenäosust selle kohta, kas ründaja võiks nüüd järgi jõuda võtame Poissoni tiheduse ja korrutame selle iga edusammuga mis ta võis saavutada selleks hetkeks:

$$\sum_{k=0}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} \cdot \begin{cases} (q/p)^{(z-k)} & if\ k \le z \\ 1 & if\ k > z \end{cases}$$

Korraldame ümber, et vältida jaotuse lõputu saba summeerimist...

$$1 - \sum_{k=0}^{z} \frac{\lambda^k e^{-\lambda}}{k!} \left(1 - (q/p)^{(z-k)}\right)$$

Konverteerime selle C keelde...

```
#include double AttackerSuccessProbability(double q, int z)

 {
        double p = 1.0 - q;
        double lambda = z * (q / p);
        double sum = 1.0; int i, k;
        for (k = 0; k <= z; k++)
        {
                double poisson = exp(-lambda);
                for (i = 1; i <= k; i++)
                        poisson *= lambda / i;
                sum -= poisson * (1 - pow(q / p, z - k));
        }
        return sum;
}
```

Jooksutades mõningaid tulemusi, näeme kuidas tõenäosus kukub eksponensiaalselt väärtusega z.

```
q=0.1
z=0 P=1.0000000
z=1 P=0.2045873
z=2 P=0.0509779
z=3 P=0.0131722
z=4 P=0.0034552
z=5 P=0.0009137
z=6 P=0.0002428
z=7 P=0.0000647
z=8 P=0.0000173
z=9 P=0.0000046
z=10 P=0.0000012

q=0.3
z=0 P=1.0000000
```

z=5 P=0.1773523
z=10 P=0.0416605
z=15 P=0.0101008
z=20 P=0.0024804
z=25 P=0.0006132
z=30 P=0.0001522
z=35 P=0.0000379
z=40 P=0.0000095
z=45 P=0.0000024
z=50 P=0.0000006

Lahendus P jaoks alla 0.1%...

P < 0.001
q=0.10 z=5
q=0.15 z=8
q=0.20 z=11
q=0.25 z=15
q=0.30 z=24
q=0.35 z=41
q=0.40 z=89
q=0.45 z=340

12. Järeldus

Oleme teinud ettepaneku ilma kolmandate osapoolteta elektrooniliste maksete süsteemi jaoks. Alustasime tavapärasest digitaalsetest allkirjadest loodud müntide raamistikuga, mis tagab küll tugeva omandiõiguse kontrolli, kuid on poolik, ilma võimaluseta ära hoida topeltkulutamist. Selle lahenduseks pakume kasutajalt-kasutajale võrku, mis kasutab töö tõendamise(Proof of work -tõlg.) algoritmi, et salvestada selline avalik tehingute ajalugu, mis muutub kiiresti ründajale ebapraktiliseks ründeobjektiks, kui ausad võrgu osalised kontrollivad suuremat osa võrgu arvutivõimsusest. Võrk on vastupidav oma struktureerimata lihtsuses. Lüüsid töötavad kõik korraga vähese koordineerimisega. Neid pole vaja identifitseerida, kuna sõnumeid ei suunata ühessegi konkreetsesse kohta ja need tuleb edastada ainult parima tahte alusel. Võrgu osalised võivad lahkuda ja uuesti ühineda võrguga igal ajal omal soovil aksepteerides töö tõendamiste ahelat, kui tõestust selle kohta mis toimus siis, kui nad ei olnud võrku ühendatud. Nad hääletavad oma protsessori võimsusega, väljendades heakskiitu kehtivatele plokkidele, töödates nende loomise kallal ja lükates tagasi kehtetud plokid, keeldudes nendega töötamast. Selle konsensusmehhanismi abil saab rakendada kõiki vajalikke reegleid ja stiimuleid.

Viited

[1] W. Dai, "b-money," http://www.weidai.com/bmoney.txt, 1998.

[2] H. Massias, X.S. Avila, and J.-J. Quisquater, "Design of a secure timestamping service with minimal trust requirements," In 20th Symposium on Information Theory in the Benelux, May 1999.

[3] S. Haber, W.S. Stornetta, "How to time-stamp a digital document," In Journal of Cryptology, vol 3, no 2, pages 99-111, 1991.

[4] D. Bayer, S. Haber, W.S. Stornetta, "Improving the efficiency and reliability of digital time-stamping," In Sequences II: Methods in Communication, Security and Computer Science, pages 329-334, 1993.

[5] S. Haber, W.S. Stornetta, "Secure names for bit-strings," In Proceedings of the 4th ACM Conference on Computer and Communications Security, pages 28-35, April 1997.

[6] A. Back, "Hashcash - a denial of service counter-measure," http://www.hashcash.org/papers/hashcash.pdf, 2002.

[7] R.C. Merkle, "Protocols for public key cryptosystems," In Proc. 1980 Symposium on Security and Privacy, IEEE Computer Society, pages 122-133, April 1980.

[8] W. Feller, "An introduction to probability theory and its applications," 1957.