Bitcoin: elektronski gotovinski sistem enakovrednih partnerjev

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

Translated in SLOVENIAN from bitcoin.org/bitcoin.pdf by bitcoin.si

Izvleček. Različica elektronskega denarja, ki bi delovala izključno v omrežju enakovrednih partnerjev, bi omogočala pošiljanje spletnih plačil neposredno od enega do drugega uporabnika brez posredovanja finančne institucije. Del rešitve predstavljajo digitalni podpisi, vendar pa se glavne koristi teh izgubijo, če je za preprečevanje dvojne porabe še vedno potrebna zaupanja vredna tretja oseba. Kot rešitev problema dvojne porabe predlagamo uporabo omrežja vsak z vsakim. To omrežje označuje transakcije s časovnimi žigi tako, da jih zgoščuje v tekočo verigo dokazov o delu, pri čemer dokaz o delu temelji na tovrstnem zgoščevanju. Tako oblikuje zapis, ki ga ni možno spremeniti brez ponovne izvedbe dokaza o delu. Najdaljša veriga ne služi le kot dokaz zaporedja pričujočih dogodkov, ampak tudi dokazuje, da izvira iz največjega bazena procesorske moči. Vse dokler večino računske moči nadzorujejo vozlišča, ki ne sodelujejo z namenom, da bi napadla omrežje, bodo ta tvorila najdaljšo verigo in prehitevala napadalce. Samo omrežje potrebuje minimalno strukturo. Sporočila se posredujejo po najboljših močeh. Vozlišča lahko po želji zapustijo in se ponovno pridružijo omrežju, pri čemer se kot dokaz tega, kaj se je zgodilo v času njihove odsotnosti, sprejme najdaljša veriga dokazov o delu.

1. Uvod

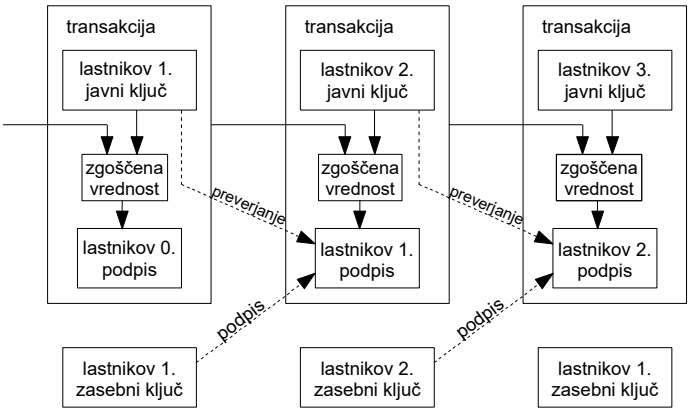
Poslovanje na internetu se zdaj že skoraj izključno zanaša na finančne institucije, ki pri obdelavi elektronskih plačil služijo kot zaupanja vredne tretje osebe. Sistem sicer za večino transakcij deluje dovolj dobro, vendar ima še vedno pomanjkljivosti, ki so neločljivo povezane z modelom, ki temelji na zaupanju. Popolnoma nepovratne transakcije v resnici niso možne, saj se finančne institucije ne morejo izogniti posredovanju pri sporih. Stroški posredovanja povečujejo transakcijske stroške, kar v praksi omejuje minimalno velikost transakcije in tako onemogoča majhne priložnostne transakcije. Stroški so zaradi izgube zmožnosti izvajanja nepovratnih plačil za nepovratne storitve, še obširnejši. Z možnostjo razveljavitve se namreč krepi potreba po zaupanju. Trgovci morajo tako biti nezaupljivi do svojih strank in od njih zahtevati več informacij, kot bi jih sicer potrebovali. Določen odstotek goljufij je sprejet kot neizogiben. Tem stroškom in plačilnim negotovostim se je mogoče izogniti z medosebno uporabo fizične valute, vendar pa ni mehanizma za plačevanje prek komunikacijskih kanalov, ki bi obstajal brez zaupanja vredne tretje osebe..

Potreben je elektronski plačilni sistem, ki bi namesto na zaupanju temeljil na kriptografskem dokazovanju. Ta bi omogočal, da bi lahko katerikoli dve osebi, ki bi to želeli, neposredno opravljali transakcije med seboj, in sicer brez potrebe po zaupanja vredni tretji osebi. Transakcije, ki jih ni praktično računsko razveljaviti, bi zaščitile prodajalce pred goljufijami, za zaščito kupcev pa bi zlahka vpeljali mehanizme rutinskih pologov. V tem članku predlagamo rešitev problema dvojne porabe z uporabo strežnika časovnih žigov, porazdeljenega v sistemu

enakovrednih partnerjev, s čimer se ustvari računski dokaz o časovnem vrstnem redu transakcij. Sistem je varen, dokler poštena vozlišča skupno nadzorujejo več procesorske moči kot katerakoli skupina sodelujočih vozlišč napadalcev.

2. Transakcije

Elektronski kovanec definiramo kot verigo digitalnih podpisov. Vsak lastnik pošlje kovanec naslednjemu z digitalnim podpisovanjem zgoščene vrednosti prejšnje transakcije in javnega ključa naslednjega lastnika, kar nato doda na konec kovanca. Prejemnik plačila lahko verificira podpise, da bi preveril verigo lastništva.



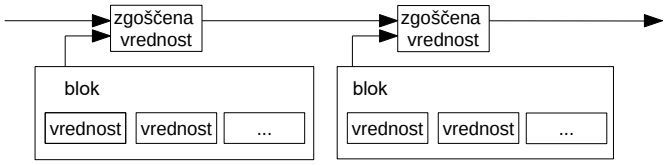
Problem je seveda v tem, ker prejemnik plačila ne more potrditi, da eden od lastnikov ni kovanca porabil dvakrat. Običajna rešitev bi bila vpeljava zaupanja vredne centralnega organa ali kovnice, ki pri vseh transakcijah preveri, da ne gre za dvojno porabo. Po vsaki transakciji se mora kovanec vrniti kovnici, da ta nato izda novega. Samo pri tistih, ki so izdani neposredno s strani kovnice, je moč zaupati, da niso bili dvakrat porabljeni. Problem pri tej rešitvi je, da je usoda celotnega denarnega sistema odvisna od podjetja, ki upravlja kovnico, ter da je treba vsako transakcijo izvesti preko le-tega, tako kot pri banki.

Potrebujemo način, da bo prejemnik plačila lahko prepričan, da prejšnji lastniki niso podpisali nobene predhodne transakcije. Za naše namene je pomembna zgolj prva transakcija, zato nas poznejši poskusi dvojne porabe ne zanimajo. Edini način, da se potrdi odsotnost neke transakcije, je, da smo seznanjeni z vsemi. V modelu, ki bazira na kovnici, je bila kovnica tista, ki je bila seznanjena z vsemi transakcijami in odločala o tem, katera je prišla prva. Da bi to dosegli brez zaupanja vredne tretje osebe, morajo biti transakcije javno objavljene [1], zato potrebujemo sistem, v katerem se udeleženci strinjajo z enotno zgodovino vrstnega reda, po katerem so bile te sprejete. Prejemnik potrebuje dokaz, da se je v trenutku vsake izmed njih večina vozlišč strinjala, da je bila ta transakcija prva.

3. Strežnik časovnih žigov

Rešitev, ki jo predlagamo, se začne pri strežniku časovnih žigov. Strežnik časovnih žigov bi deloval tako, da bi vzeli zgoščeno vrednost bloka elementov, ki bi jih označili s časovnim žigom, izračunano zgoščeno vrednost pa javno objavili, kot na primer v časopisu ali na Usenetu [2-5]. Časovni žig torej dokazuje, da so podatki v času žiga obstajali, kar je povsem očitno in zato so lahko bili vključeni v zgoščeno vrednost. Vsak časovni žig v svoji zgoščeni vrednosti vključuje prejšnjega, kar tvori verigo, kjer z vsakim naslednjim dodanim časovnim žigom le še ojača tiste

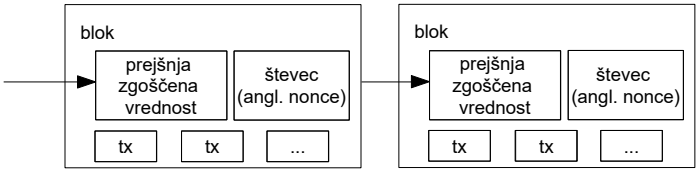
pred njim.



4. Dokaz o delu

Za implementacijo porazdeljenega strežnika časovnih žigov na osnovi omrežja enakovrednih partnerjev bomo bolj kot objave v časopisu ali na Usenetu potrebovali sistem z dokazom o delu, ki je podoben sistemu Hashcash [6] Adama Backa. Dokaz o delu vključuje iskanje vrednosti, kjer se, ko je zgoščena, na primer z zgoščevalno funkcijo SHA-256, začne z določenim številom ničelnih bitov. Povprečno zahtevano delo je eksponentno glede na število zahtevanih ničelnih bitov in ga je moč preveriti z izvedbo ene zgoščevalne funkcije.

Za naš strežnik časovnih žigov izvajamo dokaz o delu s povečevanjem števca v bloku, dokler ni najdena vrednost, ki zgoščeni vrednosti bloka vrne zahtevano število ničelnih bitov. Ko je procesorska moč porabljena tako, da zadošča dokazu o delu, blok ne more biti spremenjen brez ponovitve dela. Ker so naslednji bloki s tem povezani v verigo, bi delo za spremembo tega vključevalo ponovno delo na vseh kasnejših blokih.



Dokaz o delu rešuje tudi problem določanja zastopanosti pri večinskem odločanju. Če bi se to izvajalo tako, da bi en IP naslov štel za en glas, bi ga lahko spodkopaval kdorkoli, ki lahko pridobi več IP naslovov. Dokaz o delu pravzaprav pomeni en procesor za en glas. Večinsko odločitev zastopa najdaljša veriga, pri kateri je bilo v dokaz o delu vloženega največ truda. Če je večina procesorske moči pod nadzorom poštenih vozlišč, bo poštena veriga rasla najhitreje in prehitela vse konkurenčne verige. Za spremembo prejšnjega bloka bi moral napadalec ponoviti dokaz o delu na tem in vseh blokih za njim, nato pa še dohiteti in prehiteti delo poštenih vozlišč. V nadaljevanju bomo pokazali, da se verjetnost dohitevanja s strani počasnejšega napadalca eksponentno zmanjšuje, ko se dodajajo naslednji bloki.

Da bi kompenzirali naraščajočo hitrost strojne opreme in spreminjajoče se zanimanje za poganjanje vozlišč skozi čas, se težavnost za dokaz o delu določa s tekočim povprečjem, ki cilja povprečno število blokov na uro. Če so ti generirani prehitro, se težavnost poveča.

5. Omrežje

Koraki za poganjanje omrežja so:

1) Nove transakcije se oddajajo vsem vozliščem.
2) Vsako vozlišče nove transakcije zbira v blok.
3) Vsako vozlišče si prizadeva poiskati zahteven dokaz o delu za svoj blok.
4) Ko vozlišče najde dokaz o delu, odda blok vsem vozliščem.
5) Vozlišča sprejmejo blok le, če so vse transakcije v njem veljavne in niso že porabljene.
6) Vozlišča izrazijo svoje sprejemanje bloka z delom na tem, da bi ustvarila naslednji blok v

verigi, pri čemer uporabljajo zgoščeno vrednost sprejetega bloka kot predhodno zgoščeno vrednost.

Vozlišča vedno obravnavajo najdaljšo verigo kot pravilno in bodo nadaljevala z delom na tem, da bi jo podaljšala. Če dve vozlišči hkrati oddata dve različni verziji naslednjega bloka, lahko nekatera vozlišča najprej prejmejo eno ali drugo. V tem primeru delajo na prvi, ki so jo prejela, drugo vejo pa shranijo za primer, če ta postane daljša. Neodločen izid bo prekinjen, ko se najde naslednji dokaz o delu in ena veja postane daljša; vozlišča, ki so delala na krajši, bodo nato preklopila na daljšo.

Ni nujno, da oddane nove transakcije dosežejo vsa vozlišča. Vse dokler dosežejo dosti vozlišč, bodo kmalu prišle v blok. Oddajanje blokov je odporno tudi na izpuščena sporočila. Če vozlišče ne prejme bloka, ga bo zahtevalo naknadno, ko prejme naslednji blok in ugotovi, da je enega zgrešilo.

6. Spodbuda

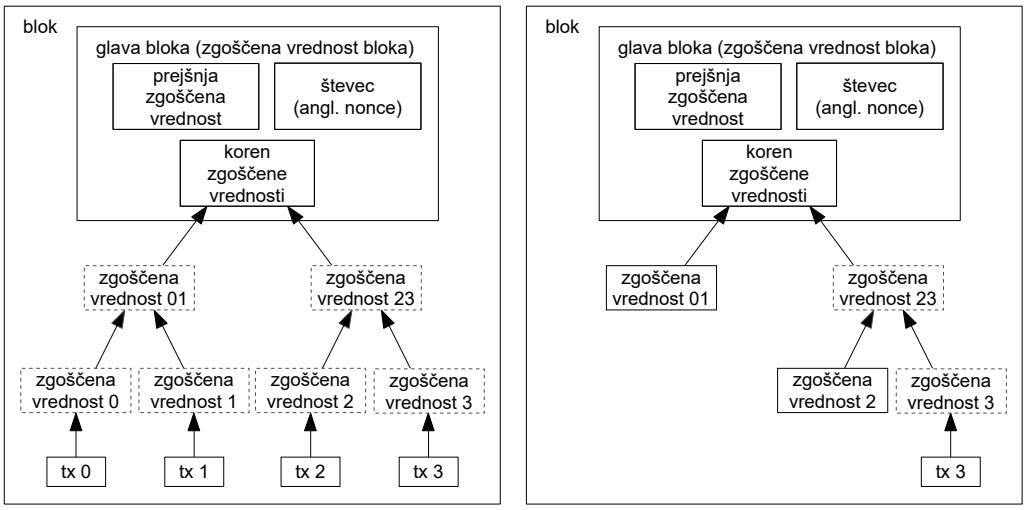
V skladu z dogovorom je prva transakcija v bloku posebna, saj ustvari nov kovanec, ki pripada kreatorju bloka. Tako se spodbuja vozlišča, da podpirajo omrežje. Obenem se s tem ustvari tudi način, s katerim se na začetku razdeli kovance v obtok, saj ni centralnega organa, ki bi jih izdajal. Redno dodajanje konstantne količine novih kovancev je primerljivo s porabo surovin zlatokopov z namenom dodajanja zlata v obtok. V našem primeru pa gre za porabo računskega časa in električne energije.

Spodbudo se lahko financira tudi s pomočjo transakcijskih stroškov. Če je izhodna vrednost transakcije manjša od vhodne, razliko predstavlja strošek, ki je dodan k vrednosti spodbude bloka, ki vsebuje transakcijo. Ko vnaprej določena količina kovancev vstopi v obtok, je lahko spodbuda v celoti sestavljena iz transakcijskih stroškov in tako postane popolnoma brez inflacije.

Spodbuda lahko pomaga motivirati vozlišča, da ostajajo poštena. Če je pohlepen napadalec sposoben zbrati več računske moči od vseh poštenih vozlišč, lahko to moč uporabi za to, da ogoljufa ljudi in ukrade nazaj svoja plačila, ali pa z njo ustvari nove kovance. Izkaže se, da se mu bolj splača igrati po pravilih, takšnih, ki mu dajo več novih kovancev kot vsem drugim skupaj, kot pa spodkopavati sistem in veljavnost lastnega bogastva.

7. Ohranjanje prostora na disku

Ko je zadnja transakcija v kovancu zakopana pod dovolj bloki, lahko porabljene transakcije pred njo zavržemo, da prihranimo prostor na disku. Da bi to olajšali, ne da poškodujemo zgoščeno vrednost bloka, so transakcije zgoščene v Merklovo drevo [7][2][5], pri čemer je v zgoščeni vrednosti bloka vključen le korenski del drevesa. Starejše bloke lahko nato zgostimo tako, da odstranimo veje drevesa. Notranjih zgoščenih vrednosti ni treba shranjevati.



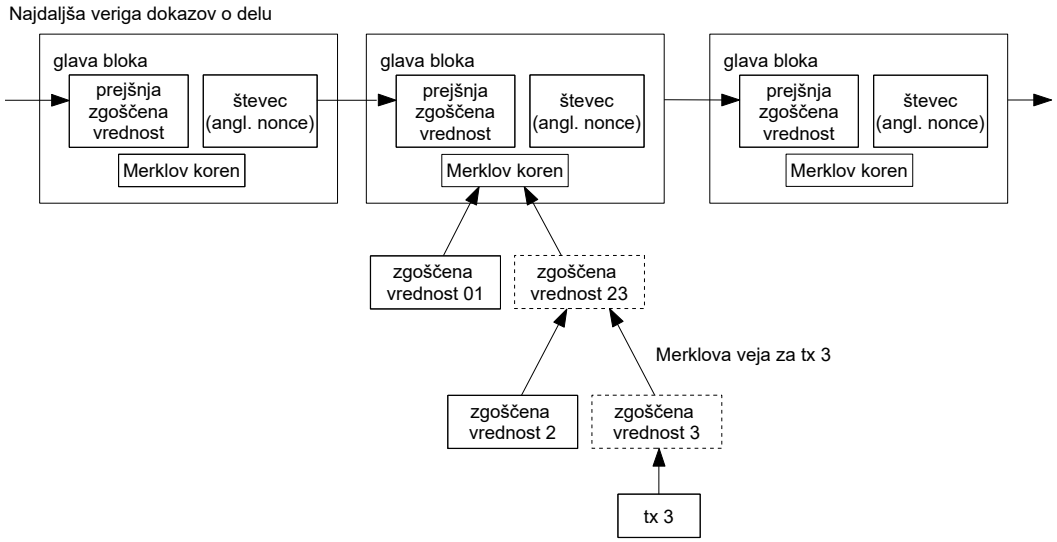
Transakcije, zgoščene na način Merklovega drevesa

Po rezanju tx 0-2 od bloka

Glava bloka brez transakcij bi bila velika približno 80 bajtov. Če predpostavimo, da se bloki generirajo vsakih 10 minut, to pomeni 80 bajtov * 6 * 24 * 365 = 4,2 MB na leto. Z računalniškimi sistemi, ki se v letu 2008 običajno prodajajo z 2 GB RAMa, in predvidevanjem Moorovega zakona, ki napoveduje trenutno rast 1,2 GB na leto, shranjevanje ne bi smelo biti težava, tudi če bi bilo treba glave blokov hraniti v pomnilniku.

8. Poenostavljeno preverjanje plačil

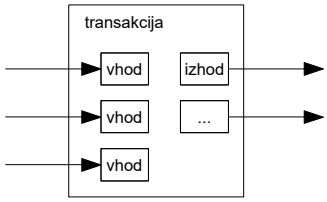
Plačila je mogoče preveriti brez poganjanja vozlišča celotnega omrežja. Uporabnik mora obdržati samo kopijo glav blokov najdaljše verige dokazov o delu. Dobi jo lahko tako, da poizveduje po omrežnih vozliščih, dokler se ne prepriča, da ima najdaljšo verigo. Prav tako pa mora pridobiti Merklovo vejo, ki povezuje transakcijo z blokom, v katerem ima časovni žig. Uporabnik transakcije ne more preveriti sam, toda če jo poveže z mestom v verigi, lahko vidi, da jo je omrežno vozlišče sprejelo. Kasneje dodani bloki pa to še dodatno potrdijo.



Kot tako je preverjanje zanesljivo, dokler omrežje nadzirajo zaupanja vredna vozlišča, postane pa bolj ranljivo, če omrežje obvladuje napadalec. Medtem ko lahko omrežna vozlišča sama preverjajo transakcije, lahko poenostavljeno metodo preslepijo napadalčeve lažne transakcije, vse dokler ima ta prevlado nad omrežjem. Ena od strategij za zaščito pred tem bi bila sprejemanje opozoril iz omrežnih vozlišč, ko bi ta zaznala neveljaven blok. S tem bi pozvala programsko opremo uporabnika, da z namenom potrditve nedoslednosti prenese celoten blok in neveljavne transakcije, o katerih je bila opozorjena. Uporabniki, ki prejemajo pogosta plačila, bodo verjetno še vedno želeli poganjati svoja vozlišča za bolj neodvisno varovanje in hitrejšo verifikacijo.

9. Združevanje in delitev vrednosti

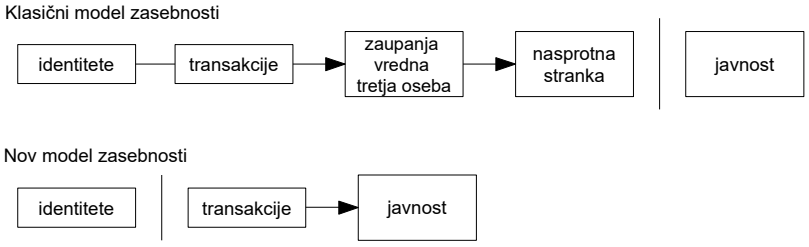
Čeprav bi bilo možno kovance obravnavati posamično, bi bilo nepraktično izvesti transakcijo za vsak nakazan delček posebej. Da bi se lahko vrednosti razdeljevale ali združevale, transakcije vsebujejo več vhodnih in izhodnih zapisov. Običajno se bo izvedel ali en sam vhodni zapis iz večje predhodne transakcije ali več vhodnih zapisov kot združitev manjših zneskov. Izhodna zapisa bosta največ dva: prvi za plačilo in drugi za vračilo morebitne razlike pošiljatelju.



Potrebno je opozoriti, da problema največje obremenitve izhoda, ki nastane, ko je transakcija odvisna od več predhodnih transakcij, te pa še od mnogih drugih pred njimi, tukaj ni. Nikoli ni treba izvleči celotne kopije zgodovine transakcije.

10. Zasebnost

Tradicionalni bančni model dosega raven zasebnosti z omejevanjem dostopa do informacij na vpletene strani in zaupanja vredno tretjo osebo. Javna objava vseh transakcij to metodo odpravlja, ampak zasebnost je še vedno mogoče ohraniti s prekinitvijo pretoka informacij drugje, in sicer z ohranjanjem anonimnosti javnih ključev. Javnost lahko vidi, da nekdo nekomu pošilja znesek, vendar nima informacij, ki bi transakcijo povezovale z določeno osebo. To je podobno ravni informacij, ki jih objavljajo borze vrednostnih papirjev, kjer sta čas in velikost posameznih poslov, "trak", javno objavljena, ni pa navedb, kdo so bile stranke.



Kot dodatno varovalko je treba za vsako transakcijo uporabiti nov par ključev, da se prepreči povezava s skupnim lastnikom. Pri transakcijah z več vhodi so povezave še vedno neizogibne,

saj te nujno razkrivajo, da so bili to vhodi istega lastnika. S tem obstaja tveganje, da zaradi povezave razkrijemo tudi druge transakcije, ki so mu pripadale, če pride do razkritja njegovega ključa.

11. Izračuni

Predstavljajmo si scenarij, ko napadalec poskuša ustvariti nadomestno verigo hitreje od poštene. Četudi to doseže, s tem sistem ne postane izpostavljen samovoljnim spremembam, kot je ustvarjanje vrednosti iz nič ali prilastitev denarja, ki mu nikoli ni pripadal. Vozlišča ne bodo sprejela neveljavne transakcije kot plačilo in poštena vozlišča nikdar ne bodo sprejela bloka, ki te vsebuje. Napadalec lahko edino poskuša spremeniti eno od lastnih transakcij, da vzame nazaj denar, ki ga je nedavno porabil.

Tekmo med pošteno in napadalčevo verigo lahko prikažemo kot binomski naključni sprehod. Uspešen dogodek pomeni podaljšanje prve za en blok, s čimer ta poveča svojo prednost za +1, neuspešen pa podaljšanje druge za en blok, kar zmanjša razkorak za -1.

Verjetnost, da napadalec nadoknadi dano zaostajanje, je primerljiva s problemom kockarjevega bankrota. Predpostavimo, da kockar z neomejenimi sredstvi začne v primanjkljaju in odigra neskončno število tekem, s čimer poskuša pokriti izgubo. Verjetnost, da bi mu to sploh kdaj uspelo, ali da bi napadalec sploh kdaj dohitel pošteno verigo, lahko izračunamo na sledeč način [8]:

p = verjetnost, da pošteno vozlišče najde naslednji blok
q = verjetnost, da napadalec najde naslednji blok
q_z = verjetnost, da bo napadalec kdaj nadoknadil zaostanek za z blokov

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

Ob predpostavki, da je p > q, se verjetnost eksponentno zmanjšuje, medtem ko se povečuje število blokov, ki jih mora napadalec nadoknaditi. Tako nima veliko možnosti in če se mu na začetku ne posreči skok naprej, te z naraščajočim zaostankom malodane izpuhtijo.

Sedaj pa poglejmo, kako dolgo mora prejemnik nove transakcije čakati, preden je lahko dovolj prepričan, da je pošiljatelj ne more spremeniti. Predpostavimo, da je pošiljatelj napadalec, ki želi prejemnika začasno prepričati, da mu je plačal, potem ko preteče nekaj časa, pa plača samemu sebi. Prejemnik bo sicer opozorjen, ko se bo to zgodilo, ampak pošiljatelj upa, da bo takrat že prepozno.

Prejemnik ustvari nov par ključev in malo pred podpisom da javni ključ pošiljatelju. To slednjemu prepreči, da bi z delom vnaprej pripravljal verigo, dokler se mu ne posreči pridobiti dovolj prednosti ter v tistem trenutku izvesti transakcijo. Ko je ta poslana, nepošteni pošiljatelj začne skrivaj delati na vzporedni verigi, ki vsebuje nadomestno različico njegove transakcije.

Prejemnik počaka, da je transakcija dodana v blok in da je z blokov povezanih za njim. Natančnega napredka napadalca sicer ne pozna, ampak če predvidevamo, da je pri nastajanju blokov poštene verige ustvaritev vsakega med njimi terjala pričakovan povprečen čas, bo napadalčev potencialni napredek Poissonova porazdelitev s predvideno vrednostjo:

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

Da bi dobili verjetnost, da napadalec še vedno lahko dohiti pošteno verigo, pomnožimo Poissonovo gostoto verjetnosti za vsako količino napredka, ki bi ga lahko naredil, z verjetnostjo, da bi lahko od tistega trenutka nadoknadil zaostanek:

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

Preureditev, da se izognemo seštevanju neskončne vrste porazdelitve …

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

Pretvorba v C kodo …

```
#include <math.h>
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;
}
```

Če izpišemo nekaj rezultatov, lahko vidimo, da verjetnost eksponentno pada s spremenljivko 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
```

Rešitve za P, manjši od 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. Zaključek

Predlagali smo sistem za elektronske transakcije, pri katerem se ni treba zanašati na zaupanje. Začeli smo z običajnim okvirom kovancev, ustvarjenih iz digitalnih podpisov, ki zagotavlja močan nadzor nad lastništvom, a je nepopoln brez načina preprečevanja dvojne porabe. Kot rešitev smo predlagali omrežje enakovrednih partnerjev, ki uporablja dokaz o delu za beleženje javne zgodovine transakcij. To v računskem smislu za napadalca hitro postane nepraktično za spreminjanje, če poštena vozlišča nadzirajo večino procesorske moči. Omrežje je robustno v svoji nestrukturirani preprostosti. Vozlišča delujejo naenkrat z le malo usklajevanja. Ni jih treba identificirati, saj sporočila niso usmerjena na določeno mesto in jih je treba le dostaviti po najboljših močeh. Vozlišča lahko omrežje po želji zapustijo in se mu ponovno pridružijo. Pri tem sprejmejo verigo dokazov o delu kot dokaz, kaj se je zgodilo, ko v omrežju niso bila prisotna. Glasujejo s svojo procesorsko močjo, pri čemer izražajo sprejemanje veljavnih blokov z vlaganjem dela v dodajanje novih blokov, neveljavne bloke pa zavračajo tako, da ne delajo na njih. S tem mehanizmom soglasja se lahko izvajajo vsa potrebna pravila in spodbude.

Literatura

[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," v *20th Symposium on Information Theory in the Benelux*, maj 1999.

[3] S. Haber, W.S. Stornetta, "How to time-stamp a digital document," v *Journal of Cryptology*, 3. del, št. 2, str. 99-111, 1991.

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

[5] S. Haber, W.S. Stornetta, "Secure names for bit-strings," v *Proceedings of the 4th ACM Conference on Computer and Communications Security*, str. 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," v *Proc. 1980 Symposium on Security and Privacy*, IEEE Computer Society, str. 122-133, april 1980.

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

Translated in SLOVENIAN from bitcoin.org/bitcoin.pdf by bitcoin.si Bitcoin association SLO.