Bitcoin: *Peer-to-peer* elektronički gotovinski sustav

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

Ovo je hrvatski prijevod tehničkog članka Bitcoin: A Peer-to-Peer Electronic Cash System autora Satoshija Nakamota objavljenog 31. listopada 2008.

Preveo: @LuxBTC (Twitter, Medium)

Sažetak

Potpuna *peer-to-peer* (svaki sa svakim) verzija elektroničkog novca omogućila bi izravno plaćanje preko interneta između dvije stranke bez posredovanja financijske ustanove. Digitalni potpisi pružaju dio rješenja, ali glavne se prednosti gube ako je za sprečavanje duple potrošnje (*double spending*) još uvijek potrebna pouzdana treća stranka. Predlažemo rješenje problema duple potrošnje koristeći *peer-to-peer* mrežu. Mreža vremenski označuje (*timestamp*) transakcije hashiranjem u kontinuirani lanac dokaza o radu (*proof-of-work*) temeljen na hashu, stvarajući zapis koji se ne može izmijeniti bez ponovnog obavljanja dokaza o radu. Najdulji lanac ne služi samo kao dokaz tj. svjedočanstvo o slijedu događaja, već i kao dokaz da je potekao iz najvećeg skupa procesorske snage (CPU). Sve dok većinskom procesorskom snagom upravljaju računala tj. čvorovi (*nodes*) koji ne planiraju u suradnji napasti mrežu, oni će proizvoditi najdulji lanac i nadjačati napadače. Sama mreža zahtijeva minimalnu strukturu. Poruke se prenose u najboljem nastojanju, a čvorovi po volji mogu napustiti mrežu i ponovno se pridružiti mreži, prihvaćajući najdulji lanac dokaza o radu kao dokaz onoga što se dogodilo dok ih nije bilo.

1. Uvod

Trgovina putem Interneta oslanja se gotovo isključivo na financijske ustanove koje služe kao pouzdana treća stranka za obradu elektroničkih plaćanja. Iako sustav funkcionira dovoljno dobro za većinu transakcija, i dalje trpi od nerazdvojivih nedostataka modela utemeljenog na povjerenju. Potpuno nepovratne transakcije nisu moguće jer financijske ustanove ne mogu izbjeći posredovanje u sporovima. Troškovi posredovanja povećavaju troškove transakcija, ograničavaju minimalnu praktičnu veličinu transakcija i onemogućuju male usputne transakcije te postoji širi trošak u gubitku mogućnosti vršenja nepovratnih plaćanja za nepovratne usluge. Potreba za povjerenjem povećava se s mogućnošću povrata plaćanja. Trgovci moraju biti na oprezu prema svojim kupcima, tražeći od njih više informacija nego što bi inače bilo potrebno. Određeni postotak prijevara je neizbježan. Ovi troškovi i nepouzdanost plaćanja mogu se izbjeći plaćanjem osobno gotovinom, ali ne postoji mehanizam za plaćanje putem komunikacijskog kanala bez pouzdane treće stranke.

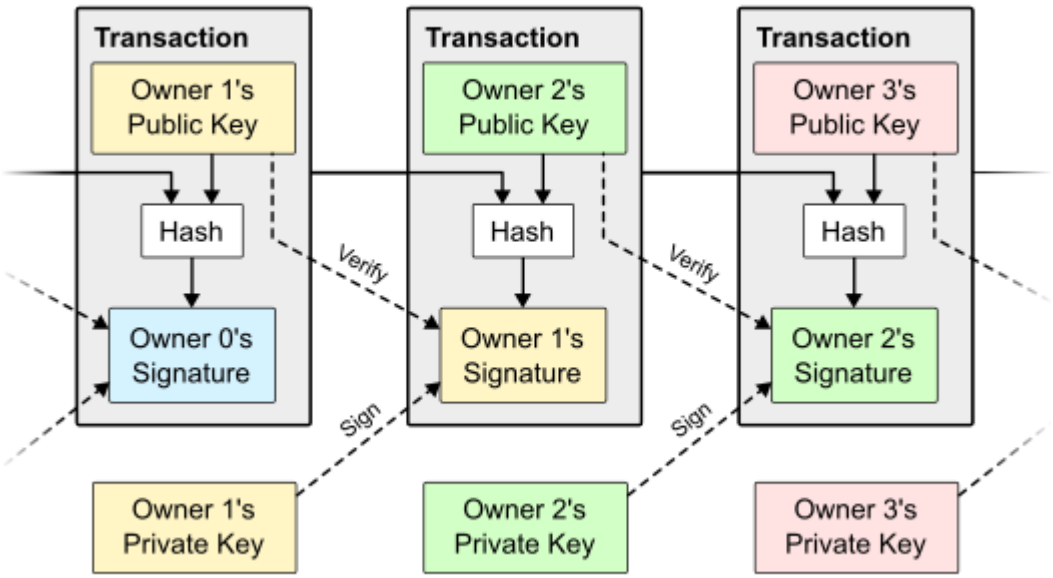
Stoga je potreban elektronički sustav plaćanja temeljen na kriptografskom dokazu umjesto na povjerenju, koji omogućava da bilo koje dvije dobrovoljne stranke posluju izravno

međusobno bez potrebe za pouzdanom trećom strankom. Transakcije koje su izračunljivo nepovratne bi zaštitile prodavače od prijevara, a rutinski založni mehanizam lako bi se mogao primijeniti za zaštitu kupaca. U ovom radu predlažemo rješenje problema duple

potrošnje putem *peer-to-peer* distribuiranog poslužitelja vremenskih oznaka za generiranje računalnog dokaza o kronološkom redoslijedu transakcija. Sustav je siguran sve dok pošteni čvorovi zajedno kontroliraju više procesorske snage od surađujuće grupe napadačkih čvorova.

2. Transakcije

Definiramo elektronički novčić kao lanac digitalnih potpisa. Svaki vlasnik prenosi novčić na sljedećeg digitalnim potpisivanjem hasha prethodne transakcije i javnog ključa sljedećeg vlasnika te ih dodaje na kraju novčića. Primatelj može provjeriti potpise kako bi potvrdio lanac vlasništva.



Problem je naravno što primatelj ne može provjeriti da jedan od vlasnika nije dvaput potrošio novac. Uobičajeno rješenje bi bilo uvođenje pouzdanog središnjeg tijela ili kovnice koja bi vršila provjeru svake transakcije na duplu potrošnju. Nakon svake transakcije, novčić se mora vratiti u kovnicu kako bi se izdao novi novčić, a samo kovanice izdane izravno iz kovnice su pouzdane da nisu duplo potrošene. Problem s ovim rješenjem je što sudbina cijelog novčanog sustava ovisi o tvrtki koja vodi kovnicu i kroz koju svaka transakcija mora proći, baš kao i u slučaju banke.

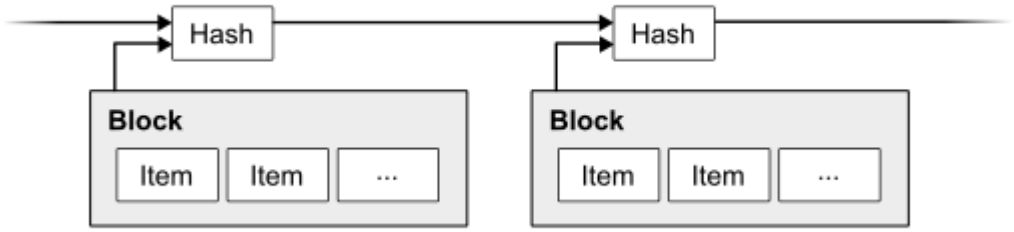
Trebamo način koji će primatelju dati do znanja da prethodni vlasnici nisu potpisali nikakve ranije transakcije. Za naše potrebe, najranija transakcija je ona koja se broji, tako da nas ne zanimaju kasniji pokušaji dvostruke potrošnje. Jedini način da provjerimo izostanak transakcije je da imamo informacije o svim transakcijama. U modelu temeljenom na kovnici, kovnica je imala informacije o svim transakcijama i odlučivala je koja transakcija je prva stigla. Da bismo to postigli bez pouzdane treće stranke, transakcije moraju biti javno objavljene [1] te nam je

potreban sustav da se sudionici slože oko jedinstvene povijesti redoslijeda kojim su transakcije primljene.

Primatelj treba dokaz da se za vrijeme svake transakcije većina čvorova složila da je prva primljena.

3. Poslužitelj vremenskih oznaka

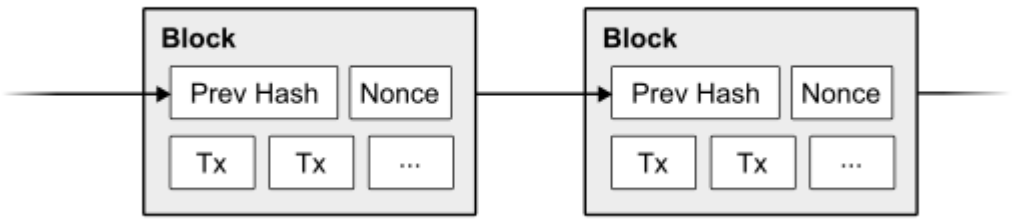
Rješenje koje predlažemo započinje s poslužiteljem vremenskih oznaka. Poslužitelj vremenskih oznaka djeluje tako da se uzme hash bloka članova kojem će se dodijeliti vremenska oznaka i objavi taj hash u primjerice novine ili Usenet [2–5]. Vremenska oznaka dokazuje da su podaci zacijelo postojali u to vrijeme, kako bi mogli ući u hash. Svaka vremenska oznaka sadrži prethodnu vremensku oznaku u svojem hashu, tvoreći lanac, pri čemu svaka dodatna vremenska oznaka ojačava one prije nje.



4. Dokaz o radu (Proof-of-Work)

Da bismo implementirali distribuirani poslužitelj vremenskih oznaka na osnovi peer-to-peer, trebat ćemo koristiti sustav dokaza o radu sličan Hashcashu Adama Backa [6] umjesto Useneta ili novinskih objava. Dokaz o radu uključuje pretraživanje vrijednosti koja nakon hashiranja, primjerice s SHA-256, tvori hash koji započinje s određenim brojem nula. Prosječna količina potrebnog rada eksponencionalna je s brojem potrebnih nula i može se provjeriti izvršavanjem jednog hasha.

Za našu mrežu vremenskih oznaka izvršavamo dokaz o radu povećanjem nonce broja u bloku dok se ne nađe vrijednost hasha tog bloka koja započinje s potrebnim brojem nula. Jednom kada se procesorska snaga utroši kako bi se zadovoljio dokaz o radu, blok se ne može izmijeniti bez da se ponovno utroši procesorska snaga. Kako se daljnji blokovi vežu na taj blok, rad potreban da se on izmijeni uključivao bi ponovno obrađivanje svih blokova nakon njega.



Dokaz o radu također rješava problem utvrđivanja zastupljenosti u većinskom odlučivanju. Ako bi se većina temeljila na principu jednog glasa po IP adresi, mogao bi je narušiti svatko tko može prikupiti više IP adresa. Dokaz o radu stoga predstavlja jedan glas po CPU. Većinsku odluku predstavlja najdulji lanac koji ujedno sadrži najveći uloženi dokaz o radu. Ako većinom procesorske snage upravljaju pošteni čvorovi, tada će pošteni lanac najbrže rasti i nadmašiti bilo koje konkurentne lance. Da bi izmijenio prethodni blok, napadač bi morao ponovno obaviti dokaz o radu za taj blok i sve blokove nakon njega, a zatim prestići rad poštenih čvorova. Kasnije ćemo pokazati da se vjerojatnost sporijeg napadača, koji pokušava sustići pošteni lanac, eksponencijalno smanjuje s dodavanjem novih blokova.

Kako bi se nadoknadio rast brzine hardvera i promjenjiv interes za pokretanjem čvora tijekom vremena, težina obavljanja dokaza o radu (*proof-of-work difficulty*) određuje se prema prosječnom broju stvorenih blokova po satu. Ako se blokovi stvaraju prebrzo, težina se povećava.

5. Mreža

Koraci za pokretanje mreže:

1. Nove transakcije emitiraju se na sve čvorove.
2. Svaki čvor sakuplja nove transakcije u blok.
3. Svaki čvor radi na pronalaženju teškog dokaza o radu za svoj blok.
4. Kad čvor pronađe dokaz o radu, on emitira blok svim čvorovima.
5. Čvorovi prihvaćaju blok samo ako su sve transakcije u njemu važeće i nisu već potrošene.
6. Čvorovi iskazuju prihvaćanje bloka tako da rade na stvaranju sljedećeg bloka u lancu uz korištenje hasha prihvaćenog bloka kao prethodni hash.

Čvorovi uvijek smatraju da je najdulji lanac ispravan i nastavit će raditi na njegovom produljenju. Ako dva čvora istodobno emitiraju različite verzije sljedećeg bloka, neki čvorovi prvo mogu primiti ili jedan ili drugi blok. U tom slučaju rade na prvom kojeg su dobili, ali zadržavaju drugu kariku lanca u slučaju da ona postane dulja. Spor će biti prekinut kada se pronađe slijedeći dokaz o radu te jedna karika lanca postane dulja; tada će se čvorovi koji su radili na drugoj karici lanca prebaciti na dulju.

Nova emitiranja transakcija ne moraju nužno doprijeti do svih čvorova. Dokle god dođu do više čvorova, ući će u blok. Emitiranje blokova tolerantno je i na propuštene poruke. Ako čvor ne primi blok, zatražit će ga kada primi sljedeći blok i shvati da je propustio jedan.

6. Poticaj

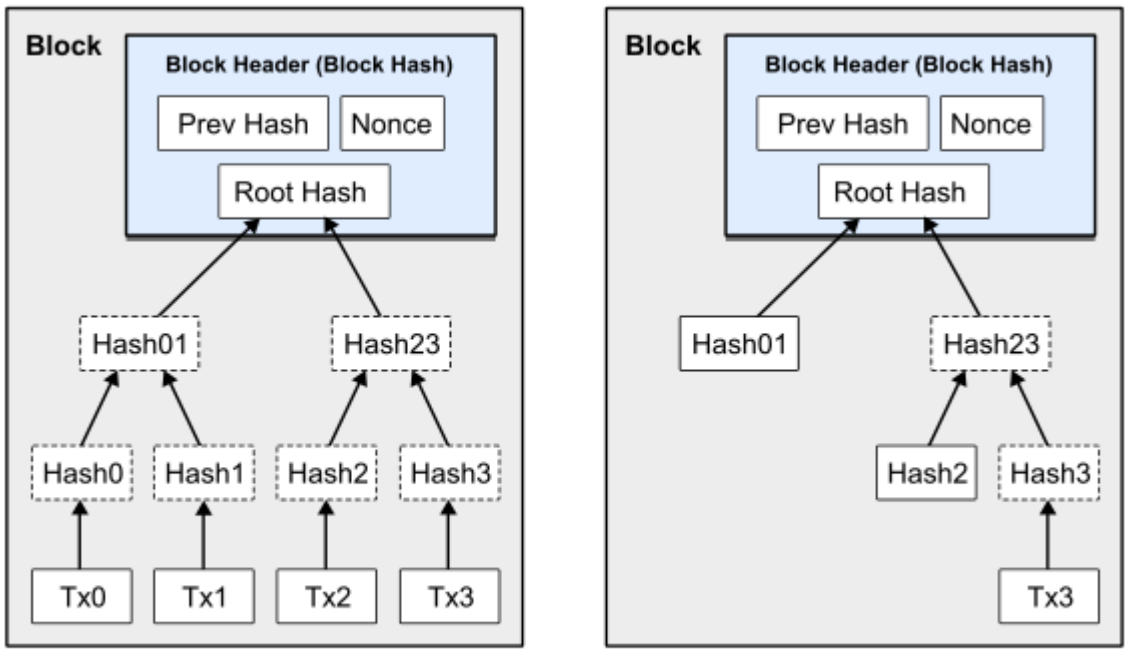
Prva transakcija u bloku je posebna po dogovoru i njome se stvara novi novčić koji pripada tvorcu bloka. Time se stvara poticaj za čvorove da sudjeluju u mreži i pruža način za početnu distribuciju novčića u optjecaj jer ne postoji središnja vlast za njihovo izdavanje. Kontinuirano dodavanje konstantne količine novih novčića analogno je rudarima zlata koji ulažu resurse kako bi dodali zlato u optjecaj. U našem slučaju ulaže se vrijeme procesorske snage čvora i električna energija.

Poticaj također može biti financiran i transakcijskim naknadama. Ako je izlazni iznos transakcije manji od njenog ulaznog iznosa, razliku čini naknada za transakciju pridodana poticajnom iznosu za blok koji sadrži transakciju (block reward). Nakon što predodređeni broj novčića uđe u optjecaj, poticaj može prijeći isključivo na naknadu za transakcije i biti potpuno bez inflacije.

Čvorovi su na taj način potaknuti da ostanu pošteni. Ako pohlepni napadač uspije prikupiti više procesorske snage od svih poštenih čvorova, morao bi birati između toga da povrati svoje uplate i time prevari ljude ili da procesorsku snagu koristi za stvaranje novih novčića. Trebao bi uvidjeti da mu je isplativije igrati po pravilima, takvim pravilima koja ga stavljaju u prednost s više novih novčića od svih ostalih zajedno, nego da potkopava sustav i vrijednost vlastitog bogatstva..

7. Oslobađanje prostora na disku

Nakon što je zadnja transakcija zakopana pod dovoljno velikim brojem blokova, potrošene transakcije prije nje se mogu odbaciti kako bi se oslobodio prostor na disku. Kako bi se to ostvarilo bez rušenja hasha bloka, transakcije se hashiraju u Merkleovo stablo [7] [2] [5] gdje je samo korijen uključen u hash bloka. Stari blokovi tada se mogu stisnuti uklanjanjem grana stabla. Unutarnje hashove nije potrebno pohranjivati.



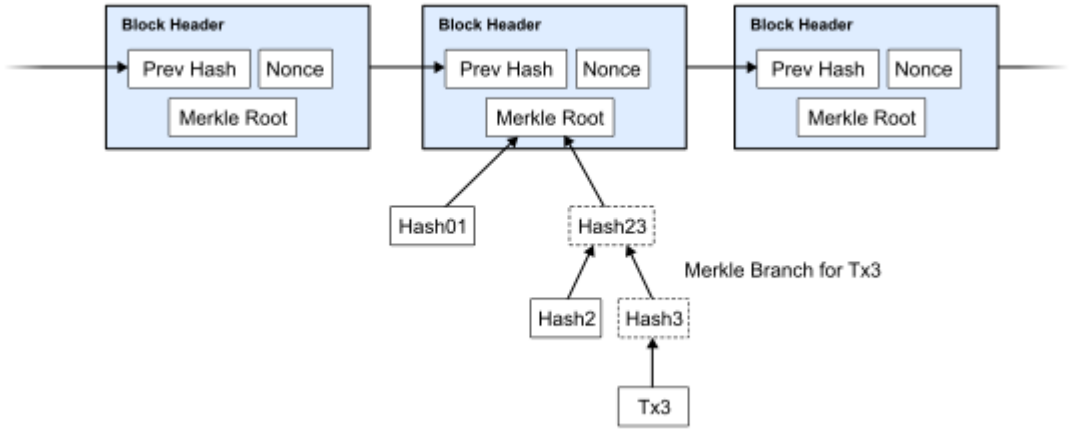
Transactions Hashed in a Merkle Tree

After Pruning Tx0-2 from the Block

Veličina zaglavlja bloka bez transakcija bila bi oko 80 bajtova. Ako pretpostavimo da se blokovi stvaraju svakih 10 minuta, 80 bajtova * 6 * 24 * 365 = 4.2 MB godišnje. Uz računalne sustave obično s 2 GB RAM-a od 2008. i Mooreov zakon koji predviđa trenutni rast od 1,2 GB godišnje, pohrana podataka ne bi trebala biti problem čak i ako zaglavlja blokova moraju biti sačuvana u memoriji računala.

8. Pojednostavljena provjera plaćanja

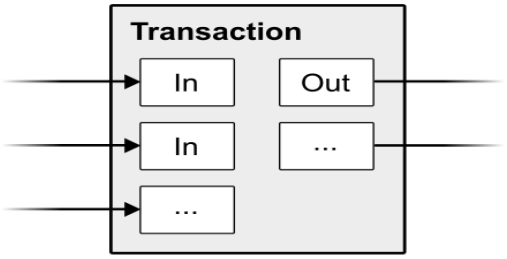
Plaćanje je moguće provjeriti bez pokretanja čitavog čvora mreže. Korisnik samo mora zadržati kopiju zaglavlja bloka najduljeg lanca, koju može saznati od čvorova mreže sve dok se ne uvjeri da ima najdulji lanac te pribaviti Merkleovu granu koja povezuje transakciju s blokom u kojem joj je dodijeljena vremenska oznaka. Ne može osobno provjeriti transakciju , ali povezujući je s mjestom u lancu može vidjeti da je prihvaćena od strane čvora mreže i da su na njen blok dodani naknadni blokovi što dalje potvrđuje da ju je mreža prihvatila.



Provjera je kao takva pouzdana dok god pošteni čvorovi kontroliraju mrežu, ali je ranjivija ako napadač nadvlada mrežom. Iako čvorovi mreže mogu sami provjeriti transakcije, napadačeve lažne transakcije mogu ovu pojednostavljenu metodu zavarati sve dok je on u stanju nadvladavati mrežom. Jedna od strategija zaštite od toga bilo bi prihvaćanje upozorenja čvorova mreže kada otkriju nevažeći blok, što će korisnikov softver prisiliti da preuzme cijeli blok i sporne transakcije kako bi potvrdio nedosljednost. Tvrtke koje primaju česte uplate vjerojatno će htjeti pokretati vlastite čvorove radi neovisnije sigurnosti i brže provjere.

9. Kombiniranje i razdvajanje vrijednosti

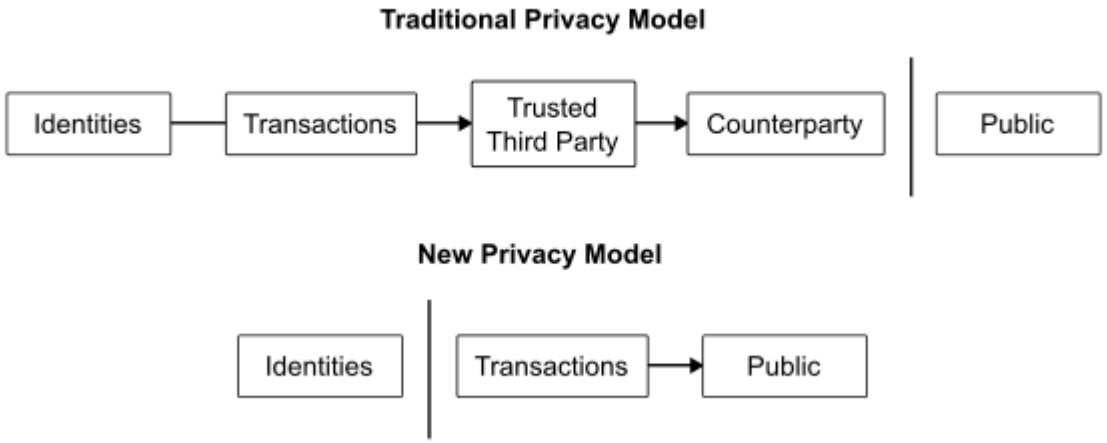
Iako bi se s novčićima moglo zasebno rukovati, bilo bi nezgrapno raditi zasebnu transakciju za svaki cent u prijenosu. Kako bi se omogućilo razdvajanje i kombiniranje vrijednosti, transakcije sadržavaju više ulaza i izlaza. Obično postoji jedan ulaz iz veće prethodne transakcije ili više ulaza koji kombiniraju manje iznose te najviše dva izlaza: jedan za isplatu i jedan koji vraća novčani ostatak, ako postoji, natrag pošiljatelju.



Treba napomenuti da izlazna lepeza *(fan-out)*, gdje transakcija ovisi o nekoliko transakcija koje ovise o mnogim drugima, ovdje nije problem. Ne postoji potreba za izvodom kompletne samostalne kopije povijesti transakcije.

10. Privatnost

Tradicionalni bankarski model postiže razinu privatnosti ograničavanjem pristupa podacima na povezane stranke te pouzdanu treću stranku. Potreba za javnim objavljivanjem svih transakcija isključuje ovu metodu, ali privatnost se i dalje može zadržati prekidom protoka podataka na drugom mjestu: anonimno čuvanje javnih ključeva (*public key*). Javno se može vidjeti da netko šalje iznos nekom drugom, ali bez informacija koje povezuju transakciju s nekim. To je slično razini informacija koje objavljuju burze, gdje se vrijeme i veličina pojedinog trgovanja objavljuju, ali bez navođenja imena stranaka.



Za dodatnu zaštitu bi se trebao koristiti novi par ključeva za svaku transakciju kako bi se spriječilo povezivanje sa zajedničkim vlasnikom. Neko je povezivanje još uvijek neizbježno, primjerice kod transakcija s više ulaza, koje nužno otkrivaju da je njihov ulaz pripadao istom vlasniku. Rizik je taj da ako je vlasnik ključa razotkriven, povezivanjem se mogu otkriti ostale transakcije koje su pripadale tom vlasniku.

11. Proračuni

Razmatramo scenarij napadača koji pokušava stvoriti alternativni lanac brže od poštenog lanca. Čak i ako se to postigne, ne ostavlja sustav ranjivim za proizvoljne promjene, poput stvaranja novca iz ničega ili uzimanja novca koji nikada nije pripadao napadaču. Pošteni čvorovi neće prihvatiti nevažeće transakcije kao uplatu te nikada neće prihvatiti blok koji ih sadrži. Napadač može pokušati promijeniti samo jednu od svojih transakcija kako bi povratio novac koji je nedavno potrošio.

Utrka između poštenog lanca i napadačkog lanca može se okarakterizirati kao Binomna slučajna šetnja. Uspješni ishod je pošteni lanac koji se produljuje za jedan blok, povećavajući svoje vodstvo za +1, a neuspješni ishod je da se lanac napadača produlji za jedan blok, smanjujući raskorak za -1.

Vjerojatnost da će napadač sustići pošteni lanac analogna je Problemu kockareve propasti. Pretpostavimo da kockar s neograničenim iznosom novca počinje s gubitkom i igra potencijalno beskonačan broj pokušaja kako bi pokušao doći na nulu. Možemo izračunati vjerojatnost za povratak na nulu ili za napadača da uhvati pošten lanac, kako slijedi [8]:

$\boldsymbol{p}$ = vjerojatnost da pošten čvor pronađe sljedeći blok

$\boldsymbol{q}$ = vjerojatnost da napadač pronađe sljedeći blok

$\boldsymbol{q_z}$ = vjerojatnost da će napadač ikada sustići $\boldsymbol{z}$ zaostalih blokova

$$q_z = \begin{cases} 1 & if\ p \leq q \\ \left(\frac{q}{p}\right)^z & if\ p > q \end{cases}$$

Uzimajući u obzir pretpostavku da je $\boldsymbol{p} > \boldsymbol{q}$, vjerojatnost opada eksponencijalno kako se povećava broj blokova koje napadač mora sustići. S izgledima protiv njega, ako mu se u startu ne posreći, njegove šanse postaju sve manje i manje s povećanjem zaostalih blokova.

Razmotrimo sada koliko dugo primatelj nove transakcije treba čekati prije nego što bude dovoljno siguran da pošiljatelj ne može promijeniti transakciju. Pretpostavljamo da je pošiljatelj napadač koji želi uvjeriti primatelja da je nedavno izvršio uplatu, a zatim izvršiti tu uplatu sebi nakon nekog vrijeme. Primatelj će biti upozoren kada se to dogodi, ali pošiljatelj se nada da će biti prekasno.

Primatelj generira novi par ključeva i daje javni ključ pošiljatelju neposredno prije potpisivanja. Ovo sprječava pošiljatelja da unaprijed pripremi lanac blokova radeći na njemu neprekidno dok ne stekne dovoljnu prednost te u tom trenutku izvrši transakciju. Nakon slanja transakcije, nepošteni pošiljatelj započinje potajno raditi na paralelnom lancu koji sadrži alternativnu verziju njegove transakcije.

Primatelj čeka dok transakcija nije dodana u blok i dok $\boldsymbol{z}$ blokova nije povezano na taj blok. Nije siguran koliki napredak je napadač postigao, ali pod pretpostavkom da su pošteni blokovi kreirani u prosječnom očekivanom vremenu po bloku, potencijalni napredak napadača bit će Poissonova raspodjela s očekivanom vrijednošću:

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

Kako bismo dobili vjerojatnost sustizanja napadača, množimo Poissonovu gustoću, za svaki iznos napretka koji je mogao postići, s vjerojatnošću sustizanja od te pozicije:

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

Preuređujemo kako bismo izbjegli sumiranje beskonačnog niza razdiobe …

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

Pretvaramo u C kod…

```
#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;
}
```

Dobivanjem rezultata vidimo da vjerojatnost eksponencijalno opada sa ***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

Rješavamo za ***p*** manji 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. Zaključak

Predložili smo sustav elektroničkih transakcija koji se ne oslanjanja na povjerenje. Započeli smo s uobičajenim okvirom novčića nastalih iz digitalnih potpisa, koji pruža snažnu kontrolu nad vlasništvom, ali je nepotpun bez načina sprječavanja dvostruke potrošnje. Kako bismo to riješili, predložili smo *peer-to-peer* mrežu koristeći dokaz o radu da bi zabilježili javnu povijest transakcija čija izmjena napadačima brzo postaje računalno nepraktična ako pošteni čvorovi kontroliraju većinu procesorske snage. Robusnost mreže čini njena nestrukturirana jednostavnost. Svi čvorovi rade istovremeno uz malo koordinacije. Ne moraju se identificirati jer se poruke ne preusmjeravaju na neko određeno mjesto nego samo trebaju biti prenešene u najboljem nastojanju. Čvorovi mogu napustiti mrežu i ponovno se pridružiti mreži prihvaćajući najdulji lanac dokaza o radu kao dokaz onoga što se dogodilo dok ih nije bilo. Glasaju svojom procesorskom snagom, radeći na produljenju lanca blokova izražavaju svoje prihvaćanje valjanih blokova i odbacuju nevažeće blokove odbijanjem njihove obrade. Bilo koja potrebna pravila i poticaji mogu se provesti ovim mehanizmom konsenzusa.

Reference

[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.