Bitcoin: Egy peer-to-peer elektronikus készpénzrendszer

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

Kivonat: Egy elektronikus, tisztán peer-to-peer alapú készpénz lehetővé tenné az online kifizetések küldését az érintett felek között anélkül, hogy arra egy pénzügyi intézményt kellene igénybe venni. A digitális aláírások jelentenék a megoldás egyik részét, de a fő előnyök elvesznének, ha még mindig szükség lenne a dupla költések megakadályozásához egy megbízott harmadik személyre. Mi egy olyan megoldást javasolunk, ahol a dupla költés problémája egy peer-to-peer hálózat használatával lenne kiküszöbölve. Ez a hálózat időbélyegezné a tranzakciókat úgy, hogy egy folyamatos, függvényalapú, proof-of-work láncolatba hash-eli őket egy olyan rekordot hozva létre, amit a hitelesítési algoritmus megismétlése nélkül nem lehetne megváltoztatni. A leghosszabb lánc nem csupán arra lenne bizonyíték, hogy az események/tranzakciók sorrendje ellenőrzött, hanem arra is, hogy a lánc létrehozásához a legtöbb CPU erőforrást használták fel. Addig, amíg a CPU erőforrások többségét olyan együttműködő, „igaz" csomópontok működtetik, amik nem a hálózat támadása céljából működnek együtt, addig ezek a csomópontok hozzák létre a támadóéval szemben a leghosszabb blokkláncokat. Egy ilyen hálózat csak minimális struktúrát igényel. Az üzenetek a legjobb erőfeszítés elve szerint közvetítődnek, a csomópontok tetszés szerint léphetnek ki és csatlakozhatnak újra a hálózathoz, hogy aztán elfogadják a leghosszabb proof-of-work láncot annak igazolásaképpen, ami a távollétükben történt.

1. Bevezetés

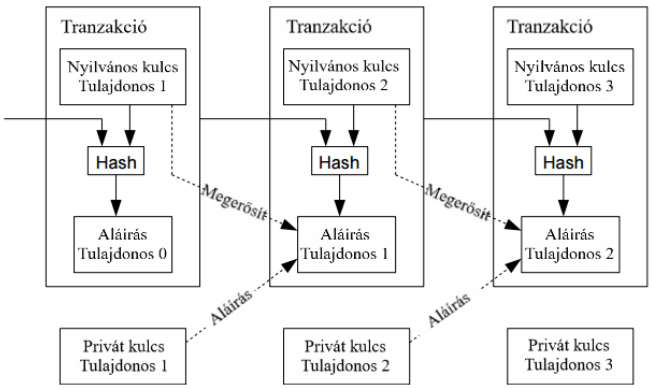
Az internetes kereskedelem az elektronikus kifizetések feldolgozásánál szinte kizárólag pénzügyi intézményekre mint megbízott harmadik személyekre hagyatkozik. Noha a rendszer a tranzakciók többségénél elég jól működik, az még mindig a bizalomalapú modell rendszerhibájától szenved. Ebben a rendszerben nem igazán lehetségesek a teljesen visszafordíthatatlan tranzakciók, ugyanis a pénzügyi intézmények nem tudják kikerülni, hogy vitákban közvetítsenek. A közvetítés eleve megdrágítja a tranzakciókat, a költségek pedig limitálják a praktikus tranzakcióméretet. Annak is ára van, hogy nem lehetséges nem visszafordítható módon fizetni nem-visszafordítható szolgáltatásokért. A visszafordíthatóság egy nagyobb bizalmi szintet igényel a felek között. A kereskedőknek ebben a rendszerben muszáj tartaniuk a vásárlóiktól, és ez több információ igénylésére ösztönzi őket, mint amennyire egyébként szükségük lenne. Ebben a rendszerben egy bizonyos csalási százalékot elkerülhetetlennek tartanak. Ezek a költségek és fizetési bizonytalanságok

elkerülhetőek akkor, ha egy fizikai fizetőeszközt használunk, de olyan mechanizmus nem létezik, amikor fizikai fizetőeszközzel kommunikációs csatornákon keresztül, megbízott harmadik fél nélkül fizetünk.

Amire szükség van, az egy olyan elektronikus fizetési rendszer, ami a bizalom helyett egy kriptográfiai bizonyítási eljáráson alapul, és ami lehetővé tenné, hogy a két érintett fél közvetlen egymásnak utaljon anélkül, hogy egy megbízott harmadik félre lenne szükség. Az olyan tranzakciók, amiknek a visszafordítása a számításikapacitás-igényt tekintve nem lenne praktikus, védenék az eladókat a csalástól, a vásárlók védelmére szolgáló zálog (escrow) mechanizmusok pedig könnyűszerrel kivitelezhetőek lennének. Ebben a fehér könyvben mi egy olyan megoldást javasolunk a dupla költés problémájára, ami egy peer-to-peer elosztású időbélyegző szervert használna a tranzakciók időbeli sorrendjét igazoló számítási bizonyítékok létrehozására. Egy ilyen rendszer addig biztonságos, amíg a "igaz" hálózati csomópontok több CPU erőforrást birtokolnak, mint a támadó csomópontok bármely együttműködő csoportja.

2. A tranzakciók

A mi definíciónk szerint egy elektronikus pénzérme nem más, mint digitális aláírások láncolata. Egy érmetulajdonos egy előző tranzakció hash-kódjának és a következő tulajdonos nyilvános kulcsának aláírásával, ezek hozzáadásával az érme végéhez juttatja el azt annak új tulajdonosához. A kedvezményezett az aláírás megerősítésével megerősíti a korábbi tulajdonosok láncolatát is.

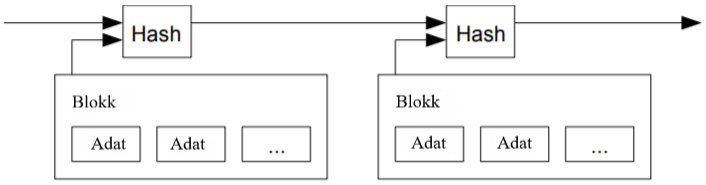


A probléma ezzel az, hogy a kedvezményezett nem tudja azt megerősíteni, hogy az előző tulajdonos nem-e költötte el kétszer ugyanazt az érmét. Az általános megoldás erre a problémára egy megbízott központi hatóság vagy pénzkibocsátó igénybevétele, amely minden egyes tranzakciót külön leellenőriz. Ilyenkor minden egyes tranzakció után az érme visszatér a pénzkibocsátóhoz, amely kibocsát egy új érmét, és csakis a kibocsátó által közvetlen kiadott érme tekinthető nem kétszer elköltöttnek. Ezzel a megoldással az a probléma, hogy a teljes pénzügyi rendszer sorsa attól a társaságtól függ, ami a pénzkibocsátót működteti, mivel minden tranzakciónak rajtuk keresztül kell történnie, akárcsak egy banknál.

Nekünk egy olyan módra van szükségünk, ahol a kedvezményezett tudja, hogy az előző tulajdonosok ugyanarra az érmére nem hagytak jóvá más korábbi tranzakciókat. A mi nézőpontunk, hogy a legrégebbbi tranzakció az, ami érvényes, ezért nem kell későbbi dupla költési kísérletekkel foglalkoznunk. Az egyetlen módja annak, hogy megerősítsük az egyéb tranzakciók nemlétét ugyanarra az érmére, ha az összes tranzakció ismert. A pénzkibocsátó-alapú modellnél a kibocsátónak tudnia kell az összes tranzakcióról és el kell döntenie, melyik érkezett be elsőnek. Hogy ezt elérhessük egy megbízott harmadik fél nélkül, a tranzakciókat nyilvánosan meg kell hirdetni [1], és egy olyan rendszerre lesz szükség, aminek a résztvevői egyetértenek abban az egyetlen tranzakciós előzményben, amin keresztül kifizetik őket. A kedvezményezettnek bizonyosságra lesz szüksége, hogy a tranzakciók ideje alatt a hálózati csomópontok többsége egyetértett abban, hogy az ő kapott érméje volt a legrégebbi.

3. Az időbélyegző szerver

Az általunk javasult megoldás kiindulópontja egy időbélyegző szerver. Egy időbélyegző szerver adatblokkok hash-eit időbélyegzi, majd ezt a hash-t széles körben nyilvánosságra hozza, mintha csak egy napilapról vagy Usenet bejegyzésről lenne szó [2-5]. Az időbélyegzés megerősíti, hogy az adatnak abban az időben nyilvánvalóan léteznie kellett ahhoz, hogy bekerülhessen a hash függvénybe. Minden időbélyeghez tartozó hash tartalmazza az előző időbélyeget, ezáltal egy lánc jön létre, ahol minden egyes újabb időbélyeg léte igazolja az azt megelőzőeket.

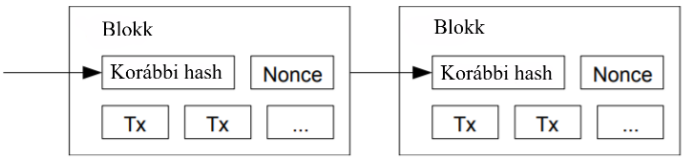


4. A proof-of-work hitelesítési algoritmus

Egy peer-to-peer alapú, megosztott időbélyegző szerver megvalósításához egy olyan proof-of-work rendszerre lesz szükségünk, mint Adam Back Hashcash-é [6], nem pedig újságokra vagy Usenet-bejegyzésekre. A proof-of-work rendszerhez tartozik egy olyan hash-elt érték beolvasása, mint a SHA-256 esetében is, a hash több nulla bittel kezdődik. A szükséges átlagos munka a szükséges zérók számát tekintve exponenciális és egyetlen hash végrehajtásával megerősíthető.

A mi időbélyegző szerverünknél a proof-of-work hitelesítést egy nonce kiegészítő változónak az eredeti hash adatsorhoz való adásával valósítjuk meg. Ez hozzáadódik a blokkhoz addig, amíg a csomópontok egyike rá nem lel arra az értékre, ami a tranzakciós blokk új hash-éhez biztosítja a szükséges számú nullát. Amint egy adott mennyiségű számítási kapacitás felhasználásra került a hitelesítési algoritmus végrehajtásához, a blokk többé nem lesz megváltoztatható anélkül, hogy a munkát ne

kelljen újból elvégezni. Mivel a későbbi blokkok hash értékei az előzőekre épülnek, egy blokk megváltoztatásával az összes azt megelőző tranzakciós blokkot újra létre kellene hozni.



A proof-of-work a többségi döntéshozatalhoz szükséges arányok meghatározásának kérdését is megoldja. Ha a szavazatok IP-cím alapján oszlanának el, ez megkerülhető lenne bárki által, aki több IP-címet is ki tud magának osztani. A proof-of-work rendszerben lényegében egy CPU jelentene egy szavazatot. A többségi döntést az a leghosszabb blokklánc jelenti, amibe a hálózat szereplői a legtöbb hitelesítési erőfeszítést fektették. Ha a számítási kapacitás többségét az "igaz" csomópontok birtokolják, az "igaz" blokklánc növekszik majd a leggyorsabban más versengő láncokat megelőzve. Egy régi blokk módosításához a támadónak újra kellene csinálni a hitelesítést abban és az azt megelőző összes blokkban, majd be kell érnie és meg kell haladnia az "igaz" csomópontok munkáját. Demonstráljuk majd, hogy annak a valószínűsége, hogy egy lassabb támadó behozza ezt a lemaradást, exponenciálisan csökken azzal, hogy a lánchoz az együttműködő csomópontok folyamatosan újabb és újabb blokkokat adnak.

A várhatóan növekvő hardversebességet és az idővel a hálózati csomópontok futtatása iránt esetleg változó érdeklődést kompenzálandó, a proof-of work nehézségét egy mozgóátlag határozza meg, ami egy óránként megoldott átlagos blokkszámot céloz. Ha az új blokkok túl gyorsan jönnek létre, a nehézség fokozódik.

5. A hálózat

A hálózat működtetésének az alábbiak a feltételei:

1) Minden hálózati csomópont megkapja az új tranzakciókat.
2) Minden csomópont egy blokkba gyűjti az új tranzakciókat.
3) Minden egyes csomópont próbál rálelni a proof-of-work adatsorra a saját blokkjához.
4) Ha egy csomópont rálel a proof-of-work adatsorra, a megoldott blokkot közvetíti a többi csomópontnak.
5) A többi csomópont csak akkor fogadja el a blokkot, ha az abban szereplő összes tranzakció érvényes és nincs benne dupla költés.
6) A csomópontok azzal fejezik ki a blokk elfogadását, hogy a láncban elkezdik a következő blokk létrehozását az elfogadott blokk hash függvényével, mint előző hash-sel.

A csomópontok mindig a leghosszabb blokkláncot fogadják el helyesnek és azon dolgoznak majd, hogy azt tovább hosszabítsák. Ha két csomópont a következő

blokkra két különböző verziót közvetít, az egyes csomópontok egyiket vagy másikat kaphatják meg elsőnek. Ebben az esetben az elsőként beérkezetten kezdenek el dolgozni, de másolatot készítenek a másikról is arra az esetre, ha az bizonyul hosszabbnak. Ez az egyenértékűség akkor szűnik meg, amikor egy csomópont rálel a proof-work-work adatsorra, és valamelyik ág hosszabb lesz a másiknál. Ekkor a csomópontok, amik addig a másik ágon dolgoztak, átállnak a hosszabbikra.

Az új tranzakciókat nem kell feltétlen az összes csomópontnak vennie. Amíg elegendő csomóponthoz közvetítik azokat, addig a blokkláncba kerülnek. A blokkok közvetítői emellett toleránsok az elhagyott üzenetekkel szemben is. Ha egy csomópont nem kap meg egy blokkot, azt akkor fogja kérni, amikor megkapja a rá következőt és észleli, hogy egyet kihagyott.

6. Ösztönző

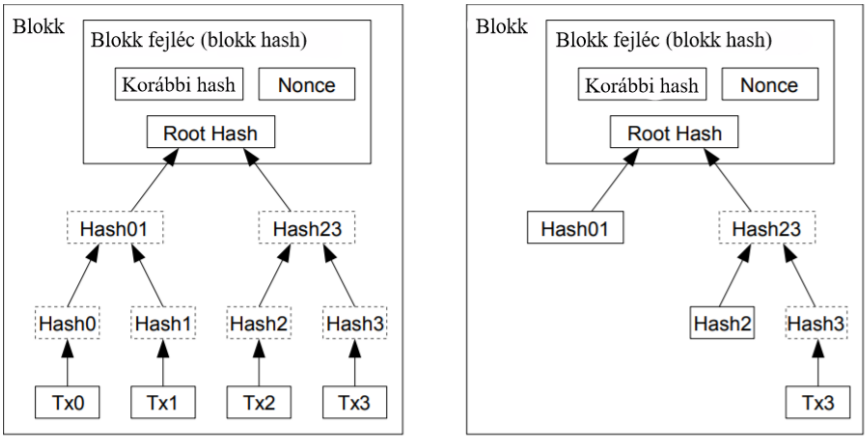
Megegyezés alapján egy blokk első tranzakciója egy olyan különleges tranzakció, ami a rendszerben egy új érmét hoz létre és amit a blokk létrehozója birtokol. Ez arra ösztönzi a csomópontokat, hogy támogassák a hálózatot, és arra is alkalmas, hogy eleinte segítse az érmék forgalomban való elosztását, hiszen azok kibocsátására itt nincsen központi szerv. Az új érmék folyamatos forgalomba kerülése hasonló ahhoz, mint amikor az aranybányászok erőforrásokat fordítanak arra, hogy több arany kerüljön forgalomba. A mi esetünben ez a ráfordított erőfeszítés elektromosság és számítási kapacitás.

Ösztönző lehet még tranzakciós díjak megállapítása. Ha egy tranzakció kimenetének az értéke kisebb, mint a bemenetéé, a különbséget egy tranzakciós díj jelenti, ami a tranzakciót tartalmazó blokk ösztönzési értékéhez adódik. Amint egy előre meghatározott mennyiségű érme forgalomba kerül, az ösztönzők rendszere teljesen átállhat a tranzakciós díjakra, és emellett teljesen inflációmentes is lehet.

Az ösztönzők arra motiválhatják a csomópontokat, hogy elfogadják a játék feltételeit. Ha egy kapzsi támadó több CPU kapacitást tud összegyűjteni, mint amennyit az összes együttműködő csomópont birtokol, választania kell, hogy becsap másokat azzal, hogy visszalopja a másnak küldött összeget, vagy használhatja ezt a kapacitást új érmék létrehozására. Úgy találhatja majd, hogy jobban megéri a szabályok szerint játszania, hiszen azok ekkora kapacitás mellett lehetővé teszik, hogy legálisan több érméhez jusson, mint az összes többi szereplő összesen, míg a rendszer aláásásával a saját vagyonának érvényességét is kikezdi.

7. A tárolókapacitás kérdése

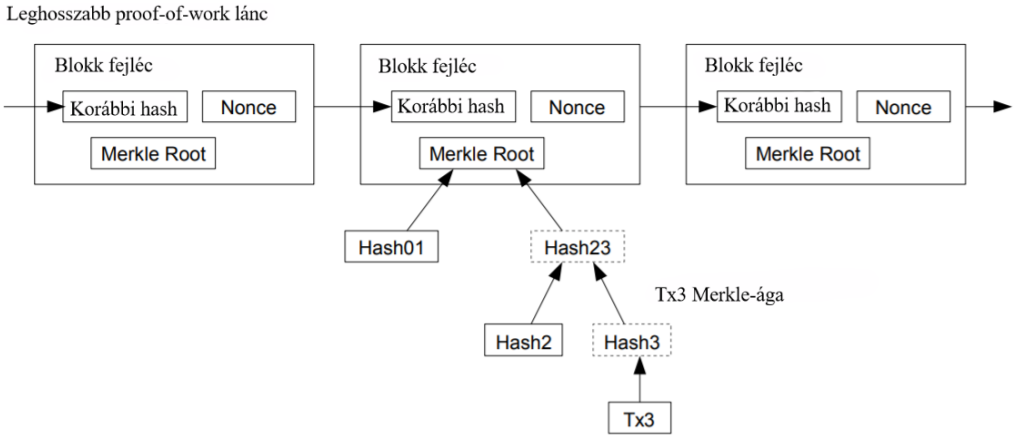
Ha egy érme utolsó tranzakcióját már elég blokk lefedi, a megelőző tranzakciókat a tárhellyel való spórolás érdekében törölni lehet. Hogy ezt megtehessük anélkül, hogy megtörnénk a blokk hash-ét, a tranzakciókat egy un. Merkle-fában [7][2][5] hash-eljük úgy, hogy a blokk hash-ébe csak a rootot vesszük bele. A régi blokkokat úgy tehetjük kompaktabbá, ahogy egy fának “lefűrészeljük” a felesleges ágait. A belső hash-ek tárolására nem lesz szükség.



Egy blokk fejléc tranzakciók nélkül körülbelül 80 byte. Ha azt feltételezzük, hogy új blokkok minden 10 percben jönnek létre, akkor 80 byte * 6 * 24 *365 = 4,2 MB évente. Tekintetbe véve, hogy 2008-ban számítógépeket tipikusan 2 GB RAM-al értékesítenek, a Moore-törvény pedig 1,2 GB-ra becsüli az éves növekedést, a tárhely még akkor sem okozhat problémát a jövőben, ha a blokk fejléceket el kell majd a memóriában tárolni.

8. Egyszerűsített fizetés-megerősítés

A kifizetések megerősítése hálózati csomópontok futtatása nélkül is lehetséges. A felhasználónak csupán egy másolatot kell megőriznie a leghosszabb proof-of-work lánc blokk fejléceiről. Ehhez addig kell lekérdeznie a csomópontokat, amíg meg nem győződik arról, hogy övé a leghosszabb lánc és meg nem kapja a Merkle rootot, ami összeköti a tranzakciót azzal a blokkal, amiben az előbbi időbélyegezve van. A felhasználó nem tudja magának ellenőriznie a tranzakciót, de a láncba való linkelésével láthatja, hogy a hálózati csomópont elfogadta azt, valamint a további blokkok jelenléte is a hálózat elfogadását erősíti meg.

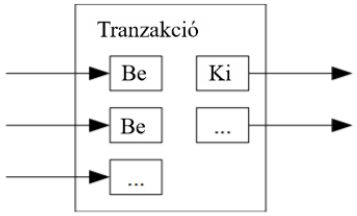


Az effajta megerősítésben mindaddig megbízhat, amíg az “igaz” csomópontok uralják a szervert. Akkor válik kérdésessé, ha a hálózatot egy támadó legyűrte. Noha a

hálózati csomópontok maguk is meg tudnak erősíteni tranzakciókat, az egyszerűsített mód becsapható a támadó által fabrikált tranzakciókkal addig, amíg a támadó erőforrásai továbbra is uralják a hálózatot. Egy lehetséges védelmi stratégia, hogy a hálózati csomópontok riadóztatják egymást, ha találnak egy érvénytelen blokkot, ami arra kényszeríti a felhasználó szoftverét, hogy letöltse a teljes blokkot és a kérdéses tranzakciókat, hogy azok érvényességét újból megerősítse. Az olyan vállalkozások, amik rendszeresen fogadnak kifizetéseket, valószínűleg saját csomópontokra akarnak majd támaszkodni a gyorsabb megerősítés és a függetlenebb biztonság érdekében.

9. Értékek kombinálása és megosztása

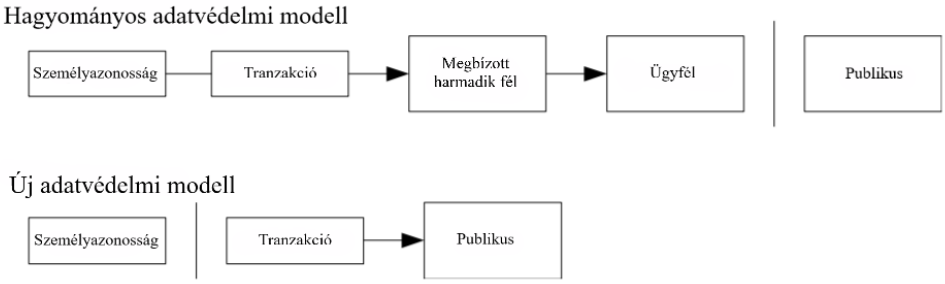
Noha elvben lehetséges az érmék külön kezelése, külön tranzakció létrehozása minden egyes centre túl körülményes lenne. Hogy egy érték megoszthatóés kombinálható legyen, a tranzakciók több bemenetet és kimenetet foglalnak magukba. Normál esetben egy nagyobb tranzakciónál egyetlen bemenet lesz vagy több olyan bemenet, ami kisebb értékeket kombinál. A legtöbb esetben két kimenet van: egy a fizetésnek, egy pedig a visszajárónak a feladó felé, ha van ilyen.



Érdemes megjegyezni, hogy a kimeneti terhelhetőség (fan-out), ahol egy tranzakció több más tranzakció függvénye és ezek maguk is mások függvényei –itt nem jelent problémát. Soha nem lesz szükség arra, hogy a tranzakciós előzményekről egy teljes, magában álló másolat készüljön.

10. Adatvédelem

A hagyományos banki modell úgy valósít meg egy adatvédelmi szintet, hogy korlátozza az információhoz való hozzáférést az érintett felek és a megbízott harmadik személy felé. Az összes tranzakció nyilvános meghirdetésének szükségessége nem teszi ezt lehetővé, de az adatvédelem még mindig fenntartható úgy, hogy az információ áramlását egy nem kívánt helyre a nyilvános kulcsok anonimitásával megtörjük. Az látható lesz, ha valaki egy bizonyos összeget küld valaki másnak, de az információ, amivel egy konkrét személyhez lehet kötni a tranzakciót, senki számára nem lesz látható. Ez hasonlít ahhoz az információs szinthez, amit a tőzsdéken használnak. Az egyes ügyletek ideje és mérete nyilvános, de az információ nem számol be az abban részt vevő felek kilétéről.



Egy további tűzfalat jelent, ha minden egyes tranzakciónál új kulcspárt használunk, hogy a privát kulcsokat se lehessen egy tulajdonoshoz kötni. A több-bemenetes tranzakciónál némi összekapcsolás azért elkerülhetetlen lesz, hiszen szükséges annak a megerősítése, hogy az összes bemenet egy tulajdonoshoz köthető. Kockázatot jelent, ha egy privát kulcs tulajdonosára fény derül, hiszen akkor a kulccsal hitelesített összes tranzakció is visszakövethetővé válik.

11. Számítások

Most egy olyan lehetőséget veszünk számításba, ahol a támadó megpróbál gyorsabban létrehozni egy alternatív blokkláncot az „igaz" lánccal szemben. Még ha ez sikerül is neki, ettől a még rendszeren nem lehet olyan önkényes változtatásokat elvégezni, mint érték létrehozása a semmiből vagy olyan pénz eltulajdonítása, ami sosem volt a támadó birtokában. A csomópontok nem fognak elfogadni fizetés gyanánt egy érvénytelen tranzakciót, az együttműködő csomópontok pedig soha nem fogadnak el olyan blokkot, ami ilyet is tartalmaz. A támadó csak a saját tranzakcióit változtathatja meg úgy, hogy visszaveszi a nemrég elköltött pénzét.

Az "igaz" és a támadó blokklánc közötti versenyt egy binomiális véletlen sétával írhatjuk le. A sikeresemény, ahol az "igaz" lánc egy blokkal meghosszabbodik, +1-el növeli annak az előnyét. A kudarcesemény, ahol támadó lánca bővül egy blokkal, vagyis az előny -1-el változik.

Annak a valószínűsége, hogy a támadó behoz egy adott hátrányt, analóg azzal a jelenséggel, amit a statisztikában a "szerencsejátékos tönkremenésének" hívnak. Tegyük fel, hogy egy szerencsejátékos végtelen számú kredit birtokában veszteséggel kezd és potenciálisan végtelen mennyiségű próbát kell tennie azért, hogy nullszaldós legyen. Ki tudjuk számolni annak a valószínűségét, hogy a szerencsejátékos valaha is nullszaldós lesz-e vagy hogy a támadó lánca valaha is beéri az "igaz" blokkláncot [8]:

p = annak a valószínűsége, hogy egy "igaz" csomópont találja meg a következő blokkot

q = annak a valószínűsége, hogy a támadó találja meg a következő blokkot

qz = annak a valószínűsége, hogy a támadó valaha is beéri az "igaz" láncot a "z" blokktól indulva

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

Ha azt feltételezzük, hogy p > q, támadó felzárkózásának a valószínűsége exponenciálisan csökken azzal, ahogy a támadó lánca által beérendő blokkok száma növekszik. A valószínűségek annyira a támadó ellen szólnak - hacsak nem sikerül egy nagyot ugrania előre az elején - hogy az esélyei egyre csak halványulnak azzal, ahogy a lemaradása növekszik.

Most azt vesszük számításba, mennyit kell az új tranzakció kedvezményezettjének várnia arra, hogy kellően biztos lehessen abban, hogy a feladó nem változtatja meg a tranzakciót. Feltételezzük, hogy ez a feladó maga a támadó, aki azt akarja elhitetni a kedvezményezettel, hogy már kifizette. Majd egy bizonyos idő elteltével a tranzakciót úgy változtatja meg, hogy visszaküldi saját magának a pénzét. A kedvezményezett erről értesül, a feladó viszont abban reménykedik, hogy ez nem történik meg időben.

A kedvezményezett létrehoz egy új kulcspárt, majd átadja a nyilvános kulcsot a feladónak közvetlen a tranzakció aláírása előtt. Ez elejét veszi annak, hogy a feladó idő előtt előkészítsen egy blokkláncot mindaddig, amíg sikerrel nem jár az előrejutásban, majd abban a pillanatban végrehajtson egy megváltoztatott tranzakciót. Ahogy a tranzakció feladásra került, a csaló elkezd titokban dolgozni egy párhuzamos láncon, abban a tranzakció egy alternatív verziójával.

A kedvezményezett addig vár, amíg a tranzakció hozzáadódik a blokkhoz és a “z” blokkok azt rendben követik. Azt nem tudja, mennyit haladt a támadó a saját láncával előre, de azt feltételezve, hogy az egyes “igaz” blokkok a várható átlagos időtartam alatt kerülnek az igaz láncba, a támadó potenciális haladása egy Poisson-eloszlást követ majd egy várható értékkel:

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

Hogy megkapjuk a támadó felzárkózási valószínűségét, a Poisson-sűrűséget megszorozzuk minden egyes haladással, amit attól a ponttól kezdve a valószínűség szerint megtehetett:

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

Ez újrarendezve, hogy elkerüljük a végtelen farok eloszlásnak az összegzését:

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

Ez C kódra átkonvertálva...

```
#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;
}
```

Néhány eredményt lefuttatva látható, hogy a valószínűség a z-vel exponenciálisan csökken.

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

Ha a P kevesebb, mint 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. Konklúzió

Ebben a fehér könyvben az elektronikus tranzakciók és készpénz egy olyan rendszerét javasoltuk, ami nem bizalmi alapú. A digitalis aláírásokkal létrehozott érmék egy szokványos keretrendszerével kezdtük, ami erős kontrollt biztosít a tulajdon felett, de nem lehet teljes a dupla költés megakadályozása nélkül. Az utóbbi problémájának a megoldására egy olyan proof-of-work hitelesítési algoritmust használó peer-to-peer hálózatot javasoltunk, amit a számítási kapacitást tekintve addig nem praktikus a támadónak megváltoztatnia, amíg az “igaz” csomópontok birtokolják a CPU kapacitás nagyobb részét. A hálózat ereje a strukturálatlan egyszerűségében rejlik. A hálózati csomópontok egyszerre dolgoznak, kevés koordinációval. A csomópontok azonosítására nincs szükség, hiszen az üzenetek nem egy adott helyre vannak irányítva és a kézbesítésük legjobb erőfeszítés-alapon történik. A csomópontok kedvük szerint csatlakozhatnak a hálózathoz és hagyhatják el azt, a távollétükban keletkezett proof-of-work láncot pedig visszatértükkor érvényesnek fogadják el. A csomópontok a CPU kapacitásukkal szavaznak, az érvényes blokkok elfogadását a blokklánc meghosszabításával, az érvénytelen blokkok elutasítását az azokon való munka megtagadásával fejezik ki. Bármilyen szükséges szabály és ösztönző ezzel a konszenzusos mechanizmussal érvényesíthető.

Források

[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.