Биткойн: Директна електронна парична система

Сатоши Накамото
satoshin@gmx.com
www.bitcoin.org

Превод: Светлозар Косев

Абстракт

Една чиста версия на електронни пари, работеща на принципа на равноправни участници (peer-to-peer), би позволила онлайн плащанията да се изпращат направо от един човек на друг, без да е необходимо да минават през финансова институция. Електронните подписи предлагат частично решение, но основните предимства изчезват, ако все пак се нуждаем от доверена трета страна, която да предотвратява двойното харчене. Ние предлагаме решение на проблема с двойното харчене, използвайки мрежа с пряка връзка. Мрежата датира всяка транзакция, като я записва в непрекъсната верига от криптографски[1] хешове[2], базирани на доказателство за извършена работа (proof-of-work). Така се създава невъзможен за промяна запис, освен ако не се извърши отново цялата работа по доказването. Най-дългата верига служи не само като доказателство за последователността на събитията, на които сме свидетели, но и като доказателство, че произлиза от най-големия обем изчислителна мощност. Докато по-голямата част от тази мощност се контролира от възли, които не си сътрудничат с цел атака на мрежата, те ще генерират най-дългата верига и ще надделеят над нападателите. Самата мрежа изисква минимална организация. Съобщенията се разпространяват според възможностите, а възлите могат да напускат и да се присъединяват отново към мрежата по всяко време, приемайки най-дългата верига от доказателства за работа като доказателство за събитията, случили се докато са отсъствали.

1 Въведение

Търговията в интернет разчита почти изключително на финансови институции, служещи като доверени трети страни за обработка на електронни плащания. Въпреки че системата работи достатъчно добре за повечето транзакции, тя все още страда от присъщите слабости на модела, основан на доверие. Напълно необратимите транзакции не са реално възможни, тъй като финансовите институции не могат да избегнат медиацията при спорове. Цената на медиацията увеличава транзакционните разходи, ограничавайки минималния практически размер на транзакцията и отрязвайки възможността за малки,

[1] Криптографията е наука за методите за запазване на информацията в тайна и за удостоверяване на нейната автентичност.

[2] Хеш е еднопосочна функция, която преобразува входни данни с произволна дължина в изходен низ с фиксирана дължина (хеш стойност), често използван за проверка на целостта на данни или за сигурност.

случайни транзакции, и има по-широка цена от загубата на възможност за извършване на необратими плащания за необратими услуги. С възможността за отмяна, необходимостта от доверие се разпространява. Търговците трябва да бъдат предпазливи към клиентите си, като ги тормозят за повече информация, отколкото иначе биха им били необходими. Известен процент измами се приемат за неизбежни. Тези разходи и несигурност при плащането могат да бъдат избегнати лично чрез използване на физическа валута, но не съществува механизъм за извършване на плащания по комуникационен канал без доверена страна.

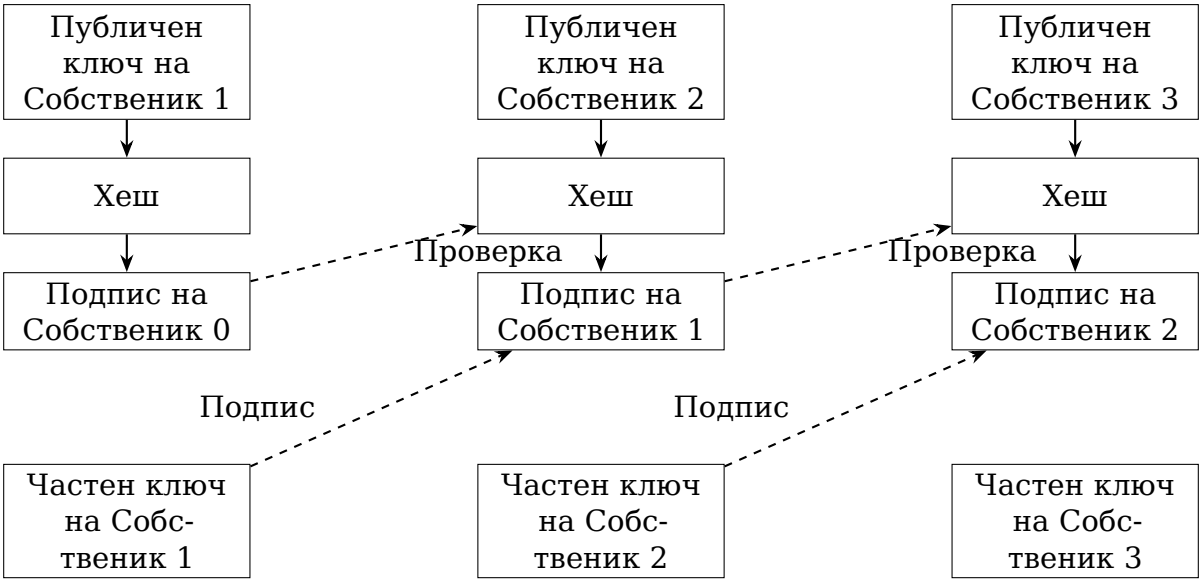
Необходима е електронна платежна система, базирана на криптографско доказателство, вместо на доверие, позволяваща на две желаещи страни да извършват директни транзакции помежду си, без да е необходима доверена трета страна. Транзакции, чието отмяна е изчислително непрактично, биха защитили продавачите от измами, а рутинни механизми за ескроу[3] биха могли лесно да бъдат внедрени за защита на купувачите. В тази статия предлагаме решение на проблема с двойното харчене, използвайки разпределен peer-to-peer сървър за времеви марки, за да генерира изчислително доказателство за хронологичния ред на транзакциите. Системата е сигурна, стига честните възли колективно да контролират повече процесорна мощност от която и да е сътрудничеща си група от атакуващи възли.

2 Транзакции

Дефинираме електронна монета като верига от цифрови подписи. Всеки собственик прехвърля монетата на следващия, като подписва цифрово хеш на предишната транзакция и публичния ключ[4] на следващия собственик и ги добавя в края на монетата. Получателят може да провери подписите, за да потвърди веригата на собственост.

[3] Ескроу (от англ. escrow) е правно и финансово споразумение, при което актив (пари, ценни книжа, документ) се държи от неутрална трета страна (ескроу агент) от името на две други страни, които са в процес на сделка. Агентът освобождава актива на бенефициента само след като бъдат изпълнени предварително определени условия. Целта е да се осигури сигурност и да се намали рискът от измама и за двете страни по сделката.

[4] Криптографски ключ, който може да бъде свободно разпространяван и използван от всеки за криптиране на съобщения или за проверка на цифрови подписи. Той е част от двойка ключове (публичен и частен), използвана в асиметричната криптография. Данни, криптирани с публичния ключ, могат да бъдат декриптирани само със съответния частен ключ, който се пази в тайна. По същия начин, подпис, създаден с частен ключ, може да бъде проверен за автентичност с публичния ключ.

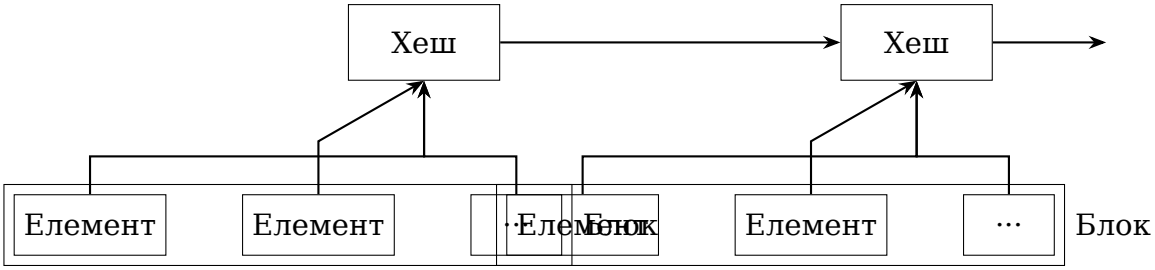


Проблемът, разбира се, е, че получателят не може да провери дали някой от собствениците не е похарчил монетата двойно. Често срещано решение е въвеждането на доверен централен орган или монетен двор, който проверява всяка транзакция за двойно харчене. След всяка транзакция монетата трябва да бъде върната в монетния двор, за да се емитира нова монета, и само монетите, емитирани директно от монетния двор, са надеждни, че няма да бъдат похарчени двойно. Проблемът с това решение е, че съдбата на цялата парична система зависи от компанията, която управлява монетния двор, като всяка транзакция трябва да премине през тях, точно както банка.

Нуждаем се от начин получателят да знае, че предишните собственици не са подписвали никакви по-ранни транзакции. За нашите цели най-ранната транзакция е тази, която се брои, така че не ни интересуват по-късни опити за двойно харчене. Единственият начин да се потвърди липсата на транзакция е да се знае за всички транзакции. В модела, базиран на монетния двор, монетният двор е бил наясно с всички транзакции и е решавал коя е пристигнала първа. За да се постигне това без доверено лице, транзакциите трябва да бъдат публично обявени [1] и се нуждаем от система, в която участниците да се споразумеят за единна история на реда, в който са били получени. Получателят се нуждае от доказателство, че по време на всяка транзакция, по-голямата част от възлите са се съгласили, че това е първата получена транзакция.

3 Сървър за времеви печати

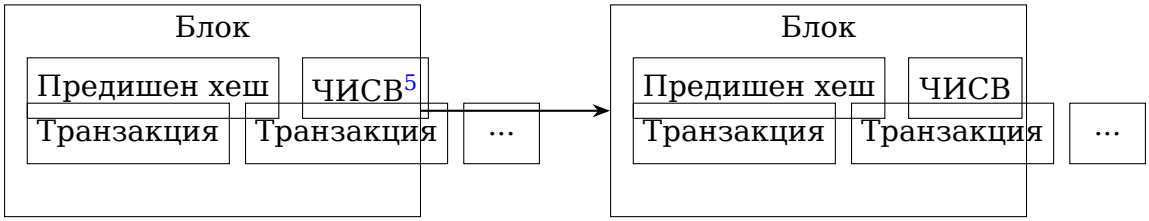
Предлаганото от нас решение започва със сървър за времеви печати. Сървърът за времеви печати работи, като взема хеш на блок от елементи, които трябва да бъдат маркирани с времеви печат, и публикува хеша широко, например във вестник или публикация в Usenet [2-5]. Времевият печат доказва, че данните е трябвало да са съществували по това време, очевидно, за да попаднат в хеша. Всеки времеви печат включва предишния времеви печат в своя хеш, образувайки верига, като всеки допълнителен времеви печат подсилва тези преди него.



4 Доказателство за извършена работа (Proof-of-Work)

За да имплементираме разпределен сървър за времеви марки на peer-to-peer база, ще трябва да използваме система за доказателство за работа, подобна на Hashcash [6] на Адам Бек, вместо публикации във вестници или Usenet. Доказателството за работа включва сканиране за стойност, която при хеширане, например с SHA-256, хешът започва с определен брой нули битове. Средната необходима работа е експоненциална в броя на необходимите нули битове и може да бъде проверена чрез изпълнение на едно хеширане.

За нашата мрежа с времеви марки, ние имплементираме доказателството за работа чрез увеличаване на nonce в блока, докато не се намери стойност, която дава на хеша на блока необходимите нули битове. След като процесорните усилия са изразходвани, за да се удовлетвори доказателството за работа, блокът не може да бъде променен без да се извърши повторно действието. Тъй като по-късни блокове се свързват след него, работата по промяната на блока ще включва повторно изпълнение на всички блокове след него.



Доказателството за работа също решава проблема с определянето на представителството при вземането на решения с мнозинство. Ако мнозинството се основаваше на един IP адрес - един глас, то би могло да бъде подкопано от всеки, който е способен да разпредели много IP адреси. Доказателството за работа е по същество един процесор - един глас. Решението на мнозинството е представено от най-дългата верига, в която са инвестирани най-големи усилия за доказателство за работа. Ако по-голямата част от мощността на процесора се контролира от честни възли, честната верига ще расте най-бързо и ще изпревари всички конкурентни вериги. За да модифицира минал блок, атакуващият ще трябва да преработи доказателството за работа на блока и всички блокове след него, а след това да настигне и надмине работата на честните възли. По-късно ще покажем, че вероятността по-бавен атакуващ да настигне намалява експоненциално с добавянето на следващи блокове.

За да се компенсира нарастващата скорост на хардуера и променливият интерес към изпълнението на възли с течение на времето, трудността на доказателството за работа се определя от пълзяща средна, насочена към среден

[5]ЧИСВ: Число, използвано само веднъж (от англ. *nonce*).

брой блокове на час. Ако те се генерират твърде бързо, трудността се увеличава.

5 Мрежа

Стъпките за стартиране на мрежата са следните:

1. Новите транзакции се излъчват до всички възли.
2. Всеки възел събира нови транзакции в блок.
3. Всеки възел работи по намирането на трудно доказателство за работа за своя блок.
4. Когато възел намери доказателство за работа, той излъчва блока до всички възли.
5. Възлите приемат блока само ако всички транзакции в него са валидни и не са вече похарчени.
6. Възлите изразяват приемането си на блока, като работят по създаването на следващия блок във веригата, използвайки хеша на приетия блок като предишния хеш.

Възлите винаги считат най-дългата верига за правилната и ще продължат да работят по нейното удължаване. Ако два възела излъчват различни версии на следващия блок едновременно, някои възли могат да получат първо едната или другата. В този случай те работят върху първия, който са получили, но запазват другия клон, в случай че стане по-дълъг. Връзката ще бъде прекъсната, когато бъде намерено следващото доказателство за работа и един клон стане по-дълъг; Възлите, които са работили по другия клон, след това ще преминат към по-дългия.

Новите транзакционни излъчвания не е задължително да достигат до всички възли. Стига да достигнат до много възли, те скоро ще влязат в блок. Блоковите излъчвания също така са толерантни към изгубени съобщения. Ако даден възел не получи блок, той ще го поиска, когато получи следващия блок и осъзнае, че е пропуснал такъв.

6 Стимул

По конвенция, първата транзакция в блок е специална транзакция, която стартира нова монета, собственост на създателя на блока. Това добавя стимул за възлите да поддържат мрежата и осигурява начин за първоначално разпределение на монетите в обращение, тъй като няма централен орган, който да ги емитира. Постоянното добавяне на постоянно количество нови монети е аналогично на това как златотърсачите изразходват ресурси, за да добавят злато в обращение. В нашия случай се изразходва процесорно време и електричество.

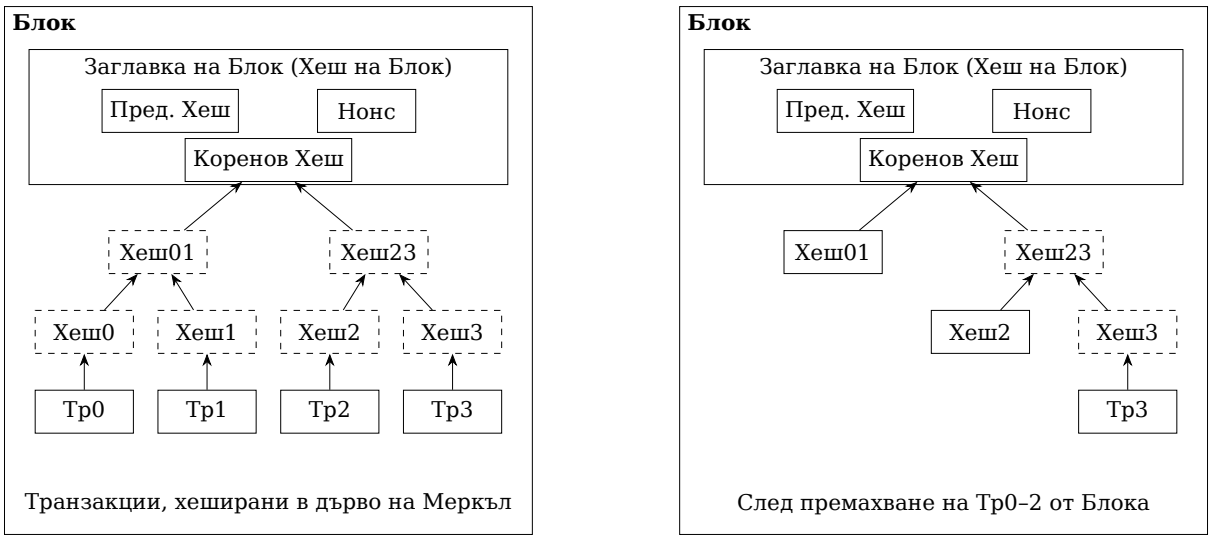
Стимулът може да бъде финансиран и с такси за транзакции. Ако изходната стойност на транзакцията е по-малка от входната ѝ стойност, разликата е такса за транзакция, която се добавя към стойността на стимула на блока, съдържащ транзакцията. След като предварително определен брой монети влязат

в обращение, стимулът може да премине изцяло към такси за транзакции и да бъде напълно без инфлация.

Стимулът може да помогне за насърчаване на възлите да останат честни. Ако алчен нападател е в състояние да събере повече процесорна мощност от всички честни възли, той ще трябва да избира между това да я използва, за да мами хора, като откраднe обратно плащанията си, или да я използва за генериране на нови монети. Той би трябвало да сметне за по-изгодно да играе по правилата, такива правила, които го облагодетелстват с повече нови монети от всички останали взети заедно, отколкото да подкопава системата и валидността на собственото си богатство.

7 Възстановяване на дисково пространство

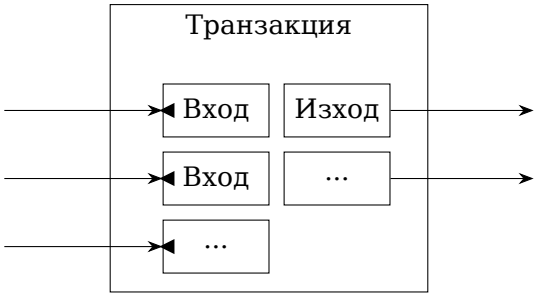
След като последната транзакция в дадена монета е скрита под достатъчно блокове, похарчените транзакции преди нея могат да бъдат изхвърлени, за да се спести място на диска. За да се улесни това, без да се нарушава хешът на блока, транзакциите се хешират в дърво на Меркъл [7][2][5], като само коренът е включен в хеша на блока. Старите блокове могат да бъдат компресирани чрез премахване на клони на дървото. Вътрешните хешове не е необходимо да се съхраняват.



По този начин, проверката е надеждна, стига честни възли да контролират мрежата, но е по-уязвима, ако мрежата е преодоляна от атакуващ. Докато мрежовите възли могат сами да проверяват транзакциите, опростеният метод може да бъде заблуден от изфабрикувани транзакции от атакуващия, стига атакуващият да може да продължи да преодолява мрежата. Една стратегия за защита срещу това би била да се приемат сигнали от мрежови възли, когато те открият невалиден блок, което да подкани софтуера на потребителя да изтегли пълния блок и да предупреди транзакциите, за да потвърди несъответствието. Фирмите, които получават чести плащания, вероятно все пак ще искат да управляват свои собствени възли за по-независима сигурност и по-бърза проверка.

8 Комбиниране и разделяне на стойност

Въпреки че би било възможно монетите да се обработват поотделно, би било тромаво да се прави отделна транзакция за всеки цент в превода. За да се позволи разделянето и комбинирането на стойността, транзакциите съдържат множество входни и изходни данни. Обикновено ще има или един вход от по-голяма предишна транзакция, или множество входни данни, комбиниращи по-малки суми, и най-много два изхода: един за плащането и един за връщане на рестото, ако има такова, обратно на подателя.



Трябва да се отбележи, че разклоняването, при което една транзакция зависи от няколко транзакции, а тези транзакции зависят от много други, не е проблем тук. Никога не е необходимо да се извлича пълно самостоятелно копие на историята на транзакцията.

9 Поверителност

Традиционният банков модел постига ниво на поверителност, като ограничава достъпа до информация до участващите страни и доверената трета страна. Необходимостта от публично обявяване на всички транзакции изключва този метод, но поверителността все пак може да се запази чрез прекъсване на потока от информация на друго място: чрез запазване на анонимността на публичните ключове. Обществеността може да види, че някой изпраща сума на някой друг, но без информация, свързваща транзакцията с когото и да било. Това е подобно на нивото на информация, публикувана от фондовите борси, където времето и размерът на отделните сделки, „лентата“, се оповестяват публично, но без да се казва кои са били страните.

Традиционният банков модел постига ниво на поверителност, като ограничава достъпа до информация до участващите страни и доверената трета страна. Необходимостта от публично обявяване на всички транзакции изключва този метод, но поверителността все пак може да се запази чрез прекъсване на потока от информация на друго място: чрез запазване на анонимността на публичните ключове. Обществеността може да види, че някой изпраща сума на някой друг, но без информация, свързваща транзакцията с когото и да било. Това е подобно на нивото на информация, публикувана от фондовите борси, където времето и размерът на отделните сделки, „лентата“, се оповестяват публично, но без да се казва кои са били страните.

Диаграма

Като допълнителна защитна стена, за всяка транзакция трябва да се използва нова двойка ключове, за да се предотврати свързването им с общ собственик.

Някои връзки все още са неизбежни при транзакции с множество входове, които задължително разкриват, че входните им данни са били собственост на един и същ собственик. Рискът е, че ако собственикът на ключ бъде разкрит, свързването може да разкрие други транзакции, които са принадлежали на същия собственик.

10 Изчисления

Разглеждаме сценария, в който атакуващ се опитва да генерира алтернативна верига по-бързо от честната верига. Дори ако това е постигнато, то не отваря системата за произволни промени, като например създаване на стойност от нищото или вземане на пари, които никога не са принадлежали на атакуващия. Възлите няма да приемат невалидна транзакция като плащане, а честните възли никога няма да приемат блок, който ги съдържа. Атакуващият може само да се опита да промени една от собствените си транзакции, за да си върне парите, които е похарчил наскоро.

Надпреварата между честната верига и веригата на атакуващия може да се характеризира като биномно случайно блуждаене. Събитието за успех е честната верига да бъде удължена с един блок, увеличавайки преднината си с +1, а събитието за неуспех е веригата на атакуващия да бъде удължена с един блок, намалявайки разликата с -1.

Вероятността атакуващият да навакса даден дефицит е аналогична на проблема с разрухата на комарджията. Да предположим, че комарджия с неограничен кредит започва с дефицит и играе потенциално безкраен брой опити, за да се опита да достигне точка на безубыточност. Можем да изчислим вероятността той някога да достигне точка на безубыточност или атакуващ някога да настигне честната верига, както следва [8]:

p = вероятност честен възел да намери следващия блок
q = вероятност атакуващият да намери следващия блок
q_z = вероятност атакуващият някога да настигне, изоставайки с z блока

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

Като се има предвид нашето предположение, че p > q, вероятността намалява експоненциално с увеличаването на броя на блоковете, които атакуващият трябва да настигне. С шансовете срещу него, ако не направи щастлив скок напред в началото, шансовете му стават изчезващо малки, докато изостава все повече.

Сега разглеждаме колко дълго трябва да чака получателят на нова транзакция, преди да е достатъчно сигурен, че подателят не може да промени транзакцията. Предполагаме, че подателят е атакуващ, който иска да накара получателя да повярва, че му е платил за известно време, след което да го превключи, за да си плати обратно, след като изтече известно време. Получателят ще бъде уведомен, когато това се случи, но подателят се надява, че ще е твърде късно.

Получателят генерира нова двойка ключове и дава публичния ключ на подателя малко преди подписването. Това предотвратява подателя да подготви верига от блокове предварително, като работи върху нея непрекъснато, докато има достатъчно късмет да стигне достатъчно напред, след което изпълнява транзакцията в този момент. След като транзакцията бъде изпратена, нечестният подател започва да работи тайно върху паралелна верига, съдържаща алтернативна версия на неговата транзакция.

Получателят изчаква, докато транзакцията бъде добавена към блок и z блока бъдат свързани след нея. Той не знае точния напредък, който е постигнал атакуващият, но ако приемем, че обработката на честните блокове е отнела средното очаквано време на блок, потенциалният напредък на атакуващия ще бъде разпределение на Поасон с очаквана стойност:

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

За да получим вероятността атакуващият все още да може да настигне сега, умножаваме плътността на Поасон за всяко количество напредък, което би могъл да постигне, по вероятността да може да настигне от тази точка:

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

Пренареждане, за да се избегне сумиране на безкрайната опашка на разпределението...

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

Пренаписване в C код...

Listing 1: Изчисляване на вероятността за успех на атакуващ

```
#include <cmath>
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;
}
```

Като изпълним някои резултати, можем да видим, че вероятността намалява експоненциално със z.

Listing 2: Изчисляване на вероятността за успех на атакуващ

```
#include <cmath>
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;
}
```

Решаване на задачата за P по-малко от 0,1%...

Listing 3: Вероятност за успех на атака (P) при различни стойности на q и z

```
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
```

Listing 4: Вероятност за успех на атака (P) при различни стойности на q и z (алтернативно представяне)

```
q=0.3
z=0   P=1.00000000
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
```

11 Заключение

Предложихме система за електронни транзакции, без да се разчита на доверие. Започнахме с обичайната рамка от монети, направени от цифрови подписи, която осигурява силен контрол върху собствеността, но е непълна без начин за предотвратяване на двойно харчене. За да решим това, предложихме peer-to-peer мрежа, използваща proof-of-work (доказателство за работа), за да записва публична история на транзакциите, която бързо става изчислително непрактична за атакуващия, ако честните възли контролират по-голямата

част от процесорната мощност. Мрежата е стабилна в своята неструктурирана простота. Възлите работят едновременно с малка координация. Не е необходимо да бъдат идентифицирани, тъй като съобщенията не се насочват към конкретно място и трябва да бъдат доставяни само на базата на максимални усилия. Възлите могат да напускат и да се присъединяват отново към мрежата по желание, приемайки веригата proof-of-work като доказателство за това какво се е случило, докато са отсъствали. Те гласуват с процесорната си мощност, изразявайки приемането си на валидни блокове, като работят по разширяването им, и отхвърлят невалидни блокове, като откажат да работят по тях. Всички необходими правила и стимули могат да бъдат наложени с този консенсусен механизъм.

12 References

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

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.

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

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.

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.

A. Back, ”Hashcash - a denial of service counter-measure,” http://www.hashcash.org/papers/hasl 2002.

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

W. Feller, ”An introduction to probability theory and its applications,” 1957.