ビットコイン：P2P 電子マネーシステム

中本 哲史

satoshi@gmx.com

www.bitcoin.co.jp

概要：純粋なP2P電子マネーによって、金融機関を通さない甲乙間の直接的オンライン取引が可能になる。電子署名は問題の一部を解決するが、依然信用できる第三者機関による二重使用予防が求めらため、その恩恵は失われる。当システムはP2P電子マネーにおける二重使用問題の解決を提案する。このネットワークは取引に、ハッシュベースの継続的なプルーフ・オブ・ワークチェーンにハッシュ値として更新日時を記録し、プルーフ・オブ・ワークをやり直さない限り変更できない履歴を作成する。最長である一連のチェーンは、取引履歴を証明するだけでなく、それがCPUパワーの最大のプールから発せられたことを証明する。大多数のCPUパワーがネットワークを攻撃していないノード（ネットワーク接続ポイント）によってコントロールされている限り最長のチェーンが作成され、攻撃者を凌ぐ。ネットワーク自体は最小限の構成でよい。メッセージは最善努力原則で送信され、ノードは自由にネットワークから離脱、再接続することができ、離脱していた間のイベントの証明として最長のプルーフ・オブ・ワークチェーンを受信する。

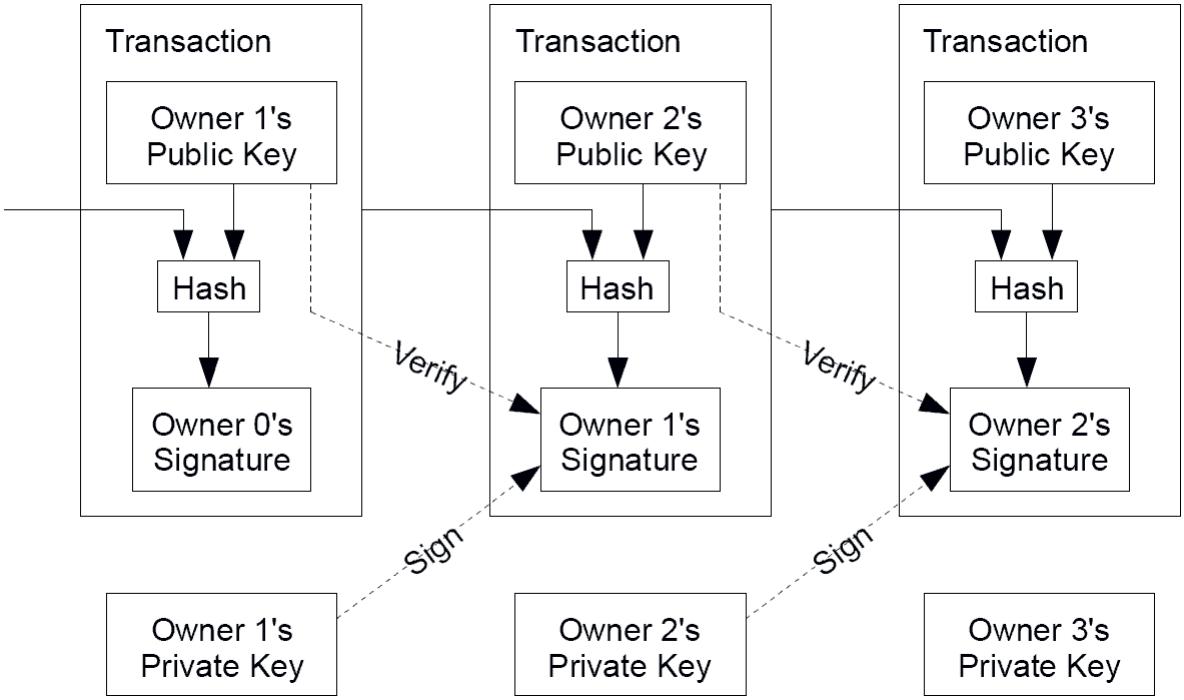
1. イントロダクション

インターネットでの商取引は、ほぼ例外なく、電子取引を処理する信用できる第三者機関としての金融機関に頼っているのが現状である。大多数の取引においてはこのシステムで十分であるものの、信頼に基づくモデルであるがゆえの弱点は残っている。金融機関は争議の仲裁を避けて通ることができないため、完全に非可逆的な取引を扱うことができない。仲裁コストが取引のコストを引き上げることで、取引規模は限定され、小額取引の可能性が失われる。また、非可逆的サービスに対する非可逆的支払いを提供することができないことによる損失はより広範にわたる。可逆的取引を扱うためには信用が問われる。商業主は顧客に対し用心深くあらねばならず、顧客から多くの情報を求める。一定の割合の詐欺は避けられないものとして受け入れられている。対個人におけるこれらの損失や支払いの不確さは有形通貨を使うことで避けられるが、第三者機関を通さずに通信チャンネル経由で支払いを可能にするメカニズムは存在していない。

必要なのは、信用ではなく暗号化された証明に基づく電子取引システムであり、これにより希望する二者が信用できる第三者機関を介さずに直接取引できるようになる。コンピュータ的に事実上非可逆的な取引は売り手を詐欺から守り、容易に実施できる習慣的なエスクロー（第三者預託）メカニズムにより買い手も守られる。この論文では、時系列取引のコンピュータ的証明を作成するP2P分散型タイムスタンプ・サーバーを用いた、二重支払い問題の解決策を提案する。本システムは、良心的なノードが集合的に、攻撃者グループのノードを上回るCPUパワーをコントロールしている限り安全である。

2. 取引

一つの電子コインは、連続するデジタル署名のチェーンと定義される。電子コインの各所有者は、直前の取引のハッシュと次の所有者のパブリック・キー（公開鍵）をデジタル署名でコインの最後に加えることにより、電子コインを次の所有者に転送する。受取人は一連の署名を検証することで、過去の所有権を検証できる。

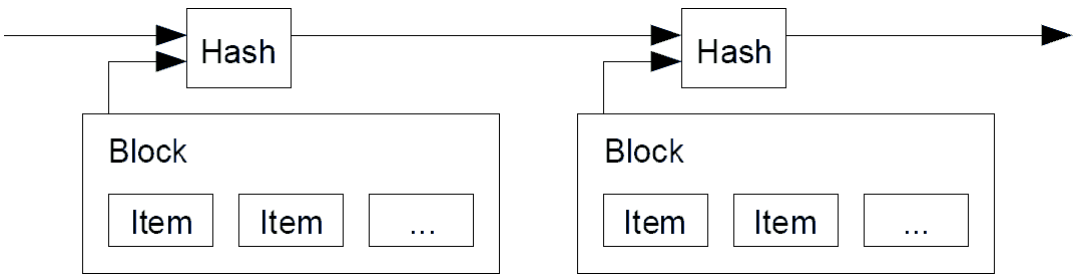


無論、問題は受取人には過去の所有者がコインを二重使用していないことを検証できないことにある。一般的な解決法は信用のおける中央機関もしくは造幣局を間に入れ、全取引を監視させることである。取引の度にコインは造幣局に戻され、新しいコインが発行され、造幣局から新しく発行されたこのコインのみが二重使用されていないものとして信用される。この解決法の問題は、全取引が造幣局を通じて行われるため、銀行と同様に造幣局を運営している企業に、金融システム全ての運命が左右されることである。

必要なのは、コインの受取人に今までの所有者らが二重署名していないことを知らせる方法である。この目的においては、最初の取引だけが論点であるので、後の二重支払いの試みについては関係のないものとする。取引がなかったことを明確にするには全取引を監視する必要がある。造幣局モデルでは造幣局が全取引を監視し取引の順番を決定していた。これを第三者機関なしに行うには、取引が公開され、参加者たちが受け取った順番の唯一の取引履歴に合意することのできるシステムが必要となる。受取人は取引毎に、取引が行われた時点で大多数のノードがそのコインが初めて使用されたことに賛同したという証明を必要とする。

3. タイムスタンプ・サーバー

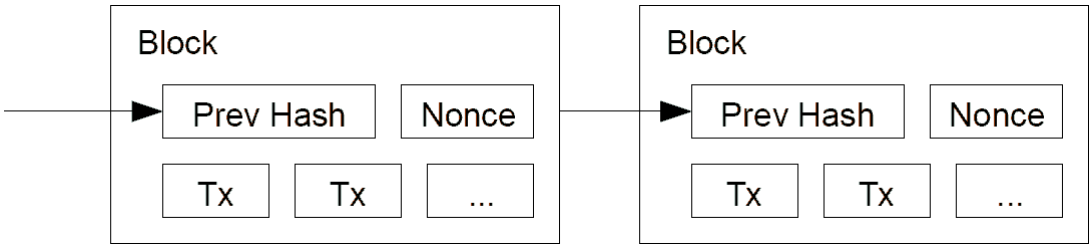
提案する解決法は、まずタイムスタンプ・サーバーから始まる。タイムスタンプ・サーバーは、タイムスタンプされる複数アイテムを含むデータブロックをハッシュとして処理し、そのハッシュを新聞やUsenetポスト[2-5]のように広範囲に公開する。タイムスタンプにより、そのデータがタイムスタンプされた時点でハッシュとなるために存在していたことが証明される。各タイムスタンプはそのハッシュの中に直前のタイムスタンプを含んでいくことでチェーンを形成し、タイムスタンプが増えるたびに以前のタイムスタンプを強化していく。



4. プルーフ・オブ・ワーク

P2Pベースで分散型サーバーを実行するには、新聞やUsenetポストというよりはアダム・バックのハッシュキャッシュ[6]に似た、プルーフ・オブ・ワークシステムを使用する必要がある。プルーフ・オブ・ワークには、例えばSHA-256のような、ハッシュ化された時にOビットの番号で始まるハッシュ値のスキャンが含まれる。通常作業に要求されるのは、必要なOビットの番号の指数関数であり、これはハッシュ一つを実行することで検証される。

我々のタイムスタンプネットワークでは、ハッシュ化の際に要求される0ビットを与える値が見つかるまでの間、データブロックにワンタイムパスワードを足すことでプルーフ・オブ・ワークを実現している。一度プルーフ・オブ・ワークを満たすべくCPUパワーが費やされると、この作業をやり直さない限りそのデータブロックを変更することはできない。その後のデータブロックもチェーン化されて後に連なるため、該当ブロックを書き換えようとするならば、それ以降の全てのブロックを書き換えなくてはならない。



このプルーフ・オブ・ワークはまた、多数決で意思決定をする際の代表をどうするかという問題を解決する。もし1IPアドレスにつき一票としたならば、多くのIPアドレスを取得できる者は誰でもシステムを乗っ取ることができてしまう。プルーフ・オブ・ワークは原則的に1CPUにつき一票である。多数決の意思決定は、最も多くのプルーフ・オブ・ワークの労力が費やされたことを示す最も長いチェーンによって表される。CPUパワーの過半数が良心的なノードによってコントロールされるとき、その良心的なチェーンは他のどのチェーンよりも早く成長する。過去のデータブロックを書き換えるためには、攻撃者はそのブロックのプルーフ・オブ・ワークだけでなくその後に続くプルーフ・オブ・ワークを書き換え、さらに良心的なチェーンに追いつき、追い越さなければならない。低速の攻撃者が良心的チェーンに追いつく可能性は、後続のブロックが追加されるごとに指数関数的に減少していくことをのちに説明する。

加速するハードウェアスピードと長期的に変動する利益レートに対応するために、プルーフ・オブ・ワーク算出の難易度は、一時間ごとのブロック数を一定の平均値に保つことを目指す平均移動によって決定される。ブロック算出のスピードが速ければ速いほど難易度が増す。

5. ネットワーク

ネットワーク実行の手順は以下の通りである:

1) 新しい取引は全ノードに送信される。
2) 各ノードが新しい取引をブロックに取り入れる。
3) 各ノードがそのブロックへのプルーフ・オブ・ワークを算出する。
4) プルーフ・オブ・ワークを見つけ次第、各ノードはそれを全ノードに告知する。
5) ノードは、ブロックに含まれる全ての取引が有効であり、以前に使われていない場合のみ、それを承認する。
6) ノードは、承認されたブロックのハッシュを直前のハッシュとして用いて、チェーンの次のブロックの作成を開始することで、ブロック承認を表明する。

ノードは常に最長のチェーンを正しいものと判断し、それをさらに延長しようとする。もし二つのノードが同時に異なる二パターンのブロックを次のブロックとして告知した場合、ノードによって受信の順番が入れ替わる可能性がある。その場合、ノードは最初に受信した方のブロックを処理するが、もう一つのブロックも保存しそちらのチェーンが長くなった場合に備えておく。次のプルーフ・オブ・ワークが発見され、どちらかのチェーンが伸びたとき、そちらが正しいチェーンと認識され、もう一つのチェーンに取り組んでいたノードはより長いチェーンに切り替える。

新しい取引の告知は必ずしも全ノードに届かなくともよい。告知が多数のノードに受信されている限り、やがてブロックに組み込まれる。ブロック告知もまたメッセージの欠落に耐えうる。ノードがブロックを受信しなかった場合、次のブロックを受信するときにそれを要求し、一つ受信していなかったことを認識する。

6. インセンティブ

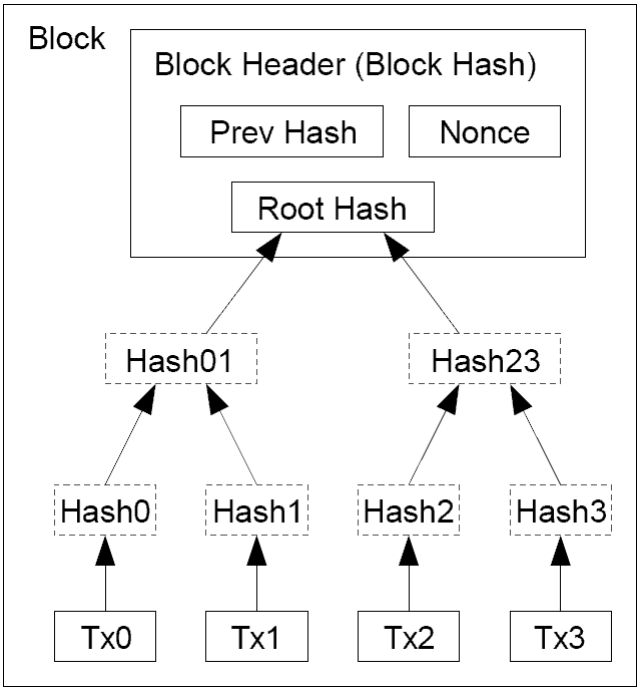
慣例により、ブロック内の最初の取引は新しいコインを始める特別な取引とされ、そのコインはブロック作成者のものとなる。これはノードにネットワークを支持するインセンティブとなると同時に、コインを発行する中央機関不在の中、最初にコインを配布する方法としても機能する。新しいコインを一定量安定して追加していくことは、金鉱労働者が働いて採金し、金の流通量を増やすことと似ている。我々の場合は、働いてるのはCPU時間と電力である。

インセンティブは、取引手数料によっても得ることができる。もしある取引でアウトプットされた価値がインプットされた価値よりも少ない場合、その差は取引手数料としてその取引を含むブロックのインセンティブに加算される。ひとたびコインの流通量が既定の数値に達するとインセンティブを取引手数料として使うことが可能になり、またコインは既定以上流通されないのでインフレからは完全に解放される。

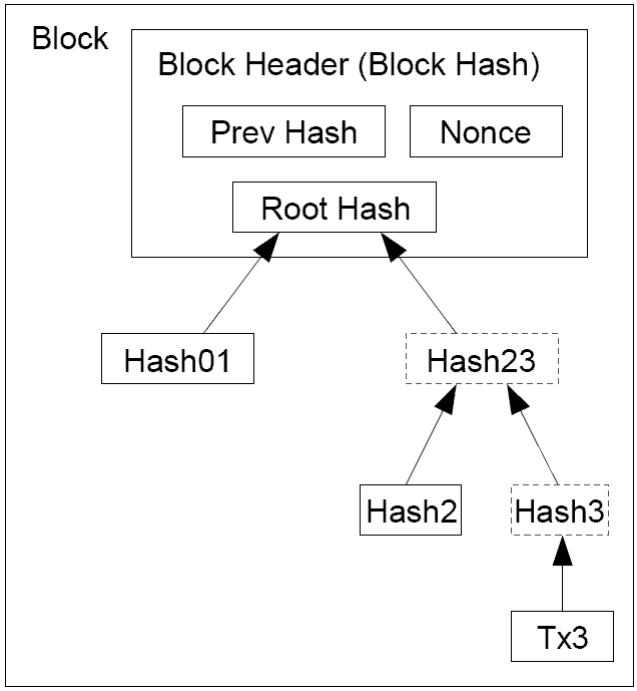
インセンティブはノードが良心的であり続ける動機となりうるだろう。もし欲深い攻撃者が良心的なノードの合計を上回るCPUパワーを作り出すことができたとして、攻撃者はそのパワーを使って、他の良心的なノードから自分の支払った金額を盗んで取り戻すか、新しいコインを作り出すかの選択を迫られることになり、おのずと自分の資産価値とそれを支えるシステムを損なうよりも、ルールに従って行動し、他の全ノードを合わせたよりも多くの新しいコインを作りだすほうが、自分の利益になると考えるだろう。

7. ディスク・スペースをリクレイムする

コインの最新の取引が十分な数のブロックに書き込まれると、それ以前の取引記録はディスク・スペースを節約するために破棄することができる。ブロックのハッシュを壊さずにこの作業を行うために、取引はそのブロックのハッシュにルートしか含まないマークル・ツリー（Merkle Tree[7][2][5]）を用いてハッシュ化される。古いブロックは、ツリーのブランチを取り除くことで軽くすることができる。インテリア・ハッシュを保存する必要はない。



マークル・ツリーを用いてハッシュ化された取引



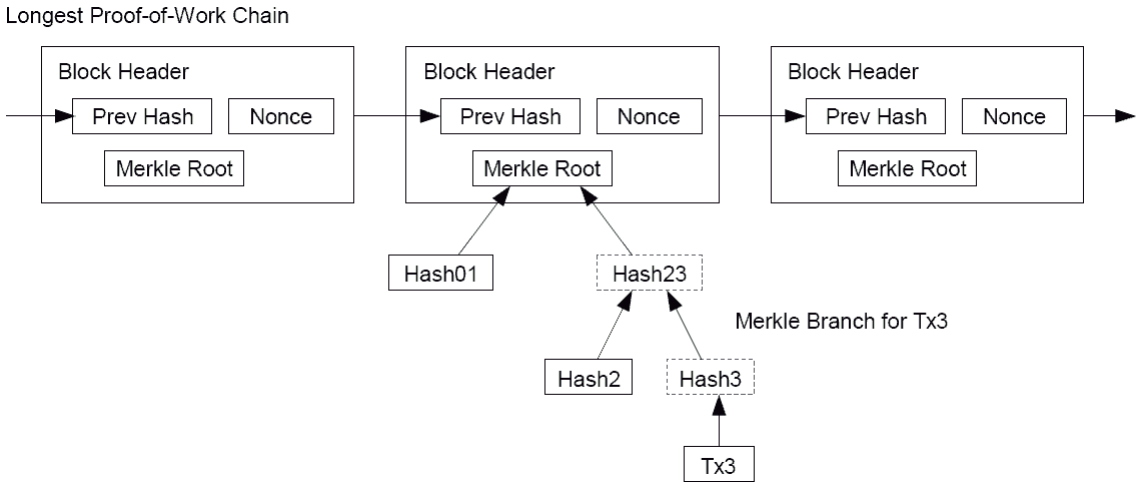
ブロックからTx0–2を取り除いた後

引なしのブロック・ヘッダーは約80 bytesである。仮に十分ごとに一つのブロックが作成されると仮定すると、80 bytes × 6 ×24 × 365 = 4.2 MB／年 となる。2008年の時点で平均的なパソコンは2 GBのRAMで売られており、ムーアの法則によると1.2 GB／年のペース

で増大すると予測されるので、ブロック・ヘッダーをメモリに保存しておく必要があってもスペースの問題はないはずである。

8. 簡易版支払い検証

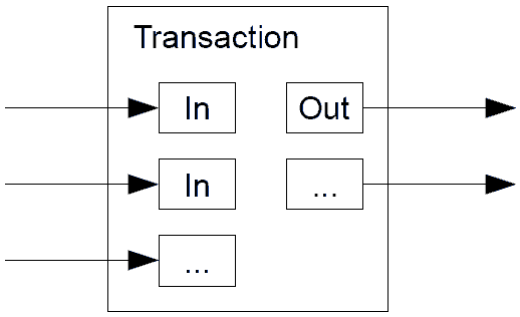
完全なネットワークノードを実行していなくとも、支払いを検証することは可能である。ユーザーは、ネットワークノードにクエリーを行って得られる最長のチェーンが含む各ブロックのブロック・ヘッダーのコピーを保存しておくだけで、そのブロックにタイムスタンプされている取引をブロックにリンクしているマークル・ブランチを得ることができるからだ。ユーザー自身が自らその取引をチェックすることはできないが、そのマークル・ブランチをチェーンにリンクすることで、ネットワークがその取引を承認済みということが確認でき、またその後にブロックが加えられていったことでさらなる確証が得られる。



このように、良心的なノードがネットワークをコントロールする限り検証は信頼のおけるものとなるが、ネットワークが攻撃者に乗っ取られた場合には脆弱になる。ネットワークは自身で取引を検証できる一方で、各ノードが行う簡易版は、攻撃者がネットワークを乗っ取り続ける限り攻撃者の偽造した取引にだまされてしまう。一つの対処法は、ネットワークが不正なブロックを感知したときに発するアラートを受信する設定にしておき、受信した場合はユーザーのソフトウェアによりブロック全体とアラートされた取引をダウンロードし、不一致を確認することである。頻繁に支払いを受け取るビジネスに関しては、より独立した安全性とスピードの速い検証のために、独自のノードを運営するほうが良いだろう。

9. 価値の結合や分割

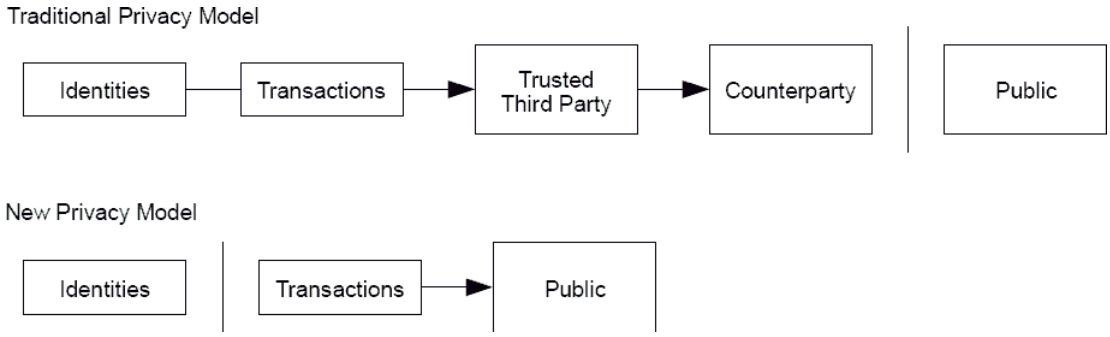
一つ一つのコインを個別に扱うことも可能な一方で、取引に使われる金額を1セントずつ個別に取引することは非常に不便だ。価値の分割や結合を可能にするために、取引には複数のインプットとアウトプットが含まれる。通常、インプットはより価値の大きい前の取引からの一つのインプットか小額のものをいくつか合わせた複数のインプットで、アウトプットは多くても二つ、次の支払いに向かうものが一つと、おつりがあればそれを支払い元に戻すものが一つである。



注目すべきは、一つの取引が複数の取引に依存し、それらの取引もまた多数の取引に依存しているファンアウトはここでは問題でないことである。取引履歴からある取引の完全に独立したコピーを抽出する必要性はあり得ないからである。

10. プライバシー

従来の銀行モデルは、情報へのアクセスを関連団体と信頼のおける第三者機関に限定することで一定レベルのプライバシーを実現している。全取引を公開する必要性はこの可能性を除外するが、情報のフローを他の箇所で分断することでプライバシーを保つことができる。パブリック・キーを匿名にするのである。誰かが他者にどれだけのコインを送っているかは公開されるが、その取引情報は誰にもリンクされていない。これは個別の取引の時間やサイズ、「テープ」は公開されても取引の当事者は明らかにされない証券取引で公表されるのと同等の情報レベルである。



追加のファイアウォールとして、同一の所有者にリンクされることを防止するために、取引は一回ごとに新しいペアのキーを用いる必要がある。複数インプットの場合はそれらのインプットが同じ所有者に所有されていることがどうしても露見するので、多少のリンクを避けることはできない。リスクは、もしキーの所有者が明らかになった場合、そのリンクにより同じ所有者が関わった他の取引も露見する可能性があることである。

11. 計算

攻撃者が正当なチェーンよりも速いスピードで偽のチェーンを作成しようとするシナリオを考えてみる。仮にそれに成功したとしても、コインを無から創り出したり攻撃者自身が所有したことのないコインを盗んだりというようにシステムを自由に操れるようになるわけではない。ノードは無効な取引を用いた支払いも無効な取引を含んだブロックも拒絶するからである。攻撃者にできるのは自身の取引記録を書き換えることで、最近支払った金額を取り返そうとすることだけである。

良心的なチェーンと攻撃者のチェーンの競争は二項酔歩（二項ランダムウォーク）と特徴づけられる。成功のイベントは良心的なチェーンが一ブロック延長してリードが一つ増えること、失敗のイベントは攻撃者のチェーンが一ブロック延長して差が一つ縮められることである。

攻撃者が与えられた遅れを取り戻す確率はギャンブラー破産の問題と似ている。無限の資金を持つギャンブラーが赤字からスタートして損益分岐点を目指して無数の賭けを行うと仮定する。彼が損益分岐点に到達する確率、もしくは攻撃者が良心的なチェーンに追いつく確率は以下のように計算することができる[8]:

P = 良心的なノードが次のブロックを見つける確率

q = 攻撃者が次のブロックを見つける確率

q_z = 攻撃者がzブロックの遅れから追いつく確率

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

p＞q というわれわれの仮定を前提とすると、追いつかなくてはならないブロックの数が増えるにしたがい確率は指数関数的に下がっていく。この分の悪い確率のもとでは、初期の段階でよほどの幸運に恵まれて突進しない限り、彼が追いつく確率は、さらに遅れをとっていく中でまずあり得ないほど小さくなっていく。

次に、新しい取引の受け手が、送り手がもう取引内容を変更できないと確信できるまでにどれだけの時間、待つ必要があるかを考える。送り手が攻撃者で、受け手に支払いを済ませたとしばらくの間信じ込ませたのちに自分に払い戻そうとしていると仮定する。その場合受け手はアラートを受信するが、送り手（攻撃者）はその時点ですでに手遅れとなっていることを望むものとする。

受け手はパブリック・キーを作成し、電子署名する直前にそれを送り手に送る。これにより、送り手が前もって偽のチェーンを用意しておきパブリック・キーが送られてきた瞬間に偽の取引を行うことを防止できる。取引が送信されるやいなや、不正な送り手（攻撃者）は密かにもう一つのバージョンの取引を含んだパラレル・チェーンの作成に取りかかる。

受け手は自分の取引がブロックに加えられ、z 個のブロックがその後にリンクされるまで待つ。受け手には攻撃者の正確な進捗は分からないが、正当なブロックが平均的な時間で作成されたと仮定すると、攻撃者の進捗は以下のポアソン分布の期待値で求められる:

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

攻撃者がこの時点で追いつくことのできる確率を求めるには、彼の行うことのできた仕事量あたりのポアソン分布密度を、その時点での追いつくことができた確率で掛ける。

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

分布の無限テールの加算を避けるために整理すると...

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

Cコードに変換すると...

```
#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;
    }
```

いくつか実行してみると、zが増えるにしたがって確率が指数関数的に下がっていくことが分かる。

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

Pが0.1%以下の時について値を求めると...

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. 結論

本論文では、信用に依存しない電子取引のシステムを提案した。電子署名で作られるコインという従来通りのフレームワークは所有権を強くコントロールを実現できるが、二重支払い防止対策なしには不完全である。その解決策として、良心的なノードがCPUパワーの過半数をコントロールする限り、プルーフ・オブ・ワークを使って記録された公開型の取引履歴を攻撃者が変えようとすることが、コンピュータ

的に加速度的に実質上実行不可能になっていくP2Pネットワークを提案した。ネットワークは非構造の単純性により堅固になっている。各ノードは同時に動作するが協調性は低い。メッセージは最善努力原則で送信すればよく、また特定の場所に転送されるわけではないのでノードが特定される必要性はない。ノードは自由にネットワークに接続、離脱でき、離脱していた間のプルーフ・オブ・ワーク・チェーンをその間の取引の証明として承認する。ノードはCPUパワーを用いて受信したチェーンへの承認・拒絶を表明し、受信したチェーンが有効であると受け入れた場合にはそれに含まれるブロックを延長することで承認を表明し、無効なブロックであればそれらの処理を拒否することで拒絶を表明する。必要なルールやインセンティブはこの合意メカニズムに従って実行できる。