Bitcoin: Sistem Uang Elektronik Peer-to-Peer

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

Abstrak. Versi uang elektronik peer-to-peer murni akan memungkinkan pembayaran daring dikirim langsung dari satu pihak ke pihak lain tanpa melalui lembaga keuangan. Tanda tangan digital menyediakan sebagian solusi, tetapi manfaat utamanya hilang jika pihak ketiga tepercaya masih diperlukan untuk mencegah pengeluaran ganda. Kami mengusulkan solusi untuk masalah pengeluaran ganda menggunakan jaringan peer-to-peer. Jaringan memberi cap waktu transaksi dengan melakukan hashing ke dalam rantai bukti kerja berbasis hash yang berkelanjutan, membentuk catatan yang tidak dapat diubah tanpa mengulang bukti kerja. Rantai terpanjang tidak hanya berfungsi sebagai bukti urutan peristiwa yang disaksikan, tetapi juga bukti bahwa peristiwa tersebut berasal dari kumpulan daya CPU terbesar. Selama sebagian besar daya CPU dikendalikan oleh node yang tidak bekerja sama untuk menyerang jaringan, mereka akan menghasilkan rantai terpanjang dan melampaui penyerang. Jaringan itu sendiri membutuhkan struktur minimal. Pesan disiarkan berdasarkan upaya terbaik, dan node dapat meninggalkan dan bergabung kembali dengan jaringan sesuai keinginan, menerima rantai bukti kerja terpanjang sebagai bukti apa yang terjadi saat mereka pergi.

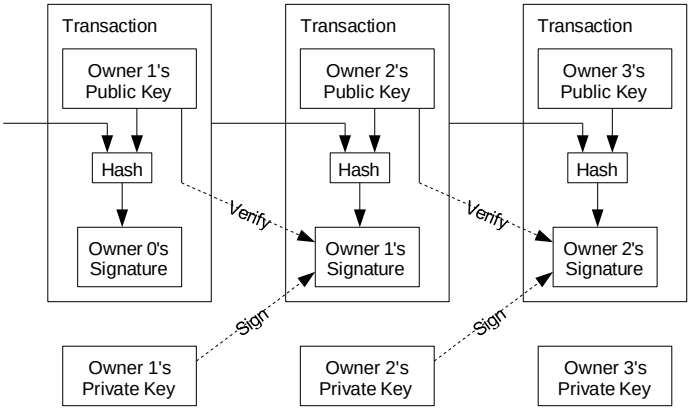
1. Perkenalan

Perdagangan di Internet hampir sepenuhnya bergantung pada lembaga keuangan yang berperan sebagai pihak ketiga terpercaya untuk memproses pembayaran elektronik. Meskipun sistem ini bekerja cukup baik untuk sebagian besar transaksi, tetap saja ada kelemahan mendasar dari model yang berbasis kepercayaan. Transaksi yang benar-benar tidak dapat dibalik (non-reversible) pada dasarnya tidak mungkin dilakukan, karena lembaga keuangan tidak bisa menghindari untuk menjadi mediator dalam sengketa. Biaya mediasi ini meningkatkan biaya transaksi, membatasi ukuran transaksi minimum yang praktis, serta menutup kemungkinan terjadinya transaksi kecil dan kasual. Lebih luas lagi, hilangnya kemampuan untuk melakukan pembayaran yang tidak dapat dibalik pada layanan yang juga tidak dapat dibalik menjadi kerugian tersendiri. Dengan adanya kemungkinan pembalikan, kebutuhan akan kepercayaan semakin meluas. Pedagang harus berhati-hati terhadap pelanggan mereka, sering kali meminta lebih banyak informasi daripada yang seharusnya dibutuhkan. Persentase tertentu dari penipuan diterima sebagai sesuatu yang tak terhindarkan. Biaya dan ketidakpastian pembayaran ini bisa dihindari ketika bertransaksi secara langsung menggunakan uang fisik, tetapi tidak ada mekanisme yang memungkinkan pembayaran melalui saluran komunikasi tanpa pihak ketiga terpercaya. Yang dibutuhkan adalah sistem pembayaran elektronik yang didasarkan pada bukti kriptografi, bukan kepercayaan, sehingga memungkinkan dua pihak yang saling bersedia untuk bertransaksi secara langsung tanpa perlu pihak ketiga terpercaya. Transaksi yang secara komputasi mustahil untuk dibalik akan melindungi penjual dari penipuan, dan mekanisme escrow rutin dapat dengan mudah diterapkan untuk melindungi pembeli. Dalam makalah ini, kami mengusulkan solusi untuk masalah double-spending dengan menggunakan server timestamp terdistribusi peer-to-peer guna menghasilkan bukti komputasi atas urutan kronologis transaksi. Sistem ini aman selama node jujur secara kolektif menguasai lebih banyak kekuatan komputasi CPU dibandingkan kelompok penyerang

yang berkolaborasi.

2. Transaksi

Kami mendefinisikan koin elektronik sebagai rangkaian tanda tangan digital. Setiap pemilik mentransfer koin ke pemilik berikutnya dengan menandatangani secara digital hash dari transaksi sebelumnya dan kunci publik pemilik berikutnya, lalu menambahkannya di akhir koin. Penerima pembayaran dapat memverifikasi tanda tangan untuk memverifikasi rangkaian kepemilikan.

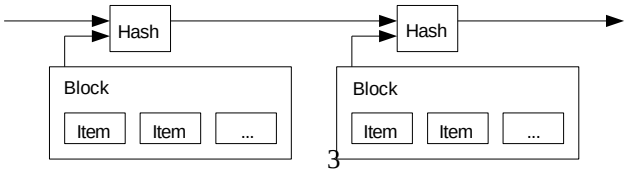


Masalahnya tentu saja penerima pembayaran tidak dapat memverifikasi bahwa salah satu pemilik tidak melakukan pengeluaran ganda pada koin tersebut. Solusi umum adalah memperkenalkan otoritas pusat tepercaya, atau percetakan uang, yang memeriksa setiap transaksi untuk pengeluaran ganda. Setelah setiap transaksi, koin harus dikembalikan ke percetakan uang untuk menerbitkan koin baru, dan hanya koin yang diterbitkan langsung dari percetakan uang yang dipercaya tidak akan melakukan pengeluaran ganda. Masalah dengan solusi ini adalah bahwa nasib seluruh sistem keuangan bergantung pada perusahaan yang menjalankan percetakan uang, dengan setiap transaksi harus melalui mereka, seperti halnya bank.

Kita memerlukan cara bagi penerima pembayaran untuk mengetahui bahwa pemilik sebelumnya tidak menandatangani transaksi sebelumnya. Untuk tujuan kita, transaksi paling awal adalah yang dihitung, jadi kita tidak peduli dengan upaya pengeluaran ganda selanjutnya. Satu-satunya cara untuk memastikan tidak adanya transaksi adalah dengan mengetahui semua transaksi. Dalam model berbasis percetakan uang, percetakan uang mengetahui semua transaksi dan memutuskan mana yang datang lebih dulu. Untuk mencapai hal ini tanpa pihak tepercaya, transaksi harus diumumkan secara publik [1], dan kita memerlukan sistem yang memungkinkan peserta menyepakati satu riwayat urutan penerimaan transaksi. Penerima pembayaran memerlukan bukti bahwa pada saat setiap transaksi, mayoritas node sepakat bahwa transaksi tersebut merupakan transaksi pertama yang diterima.

3. Server Stempel Waktu

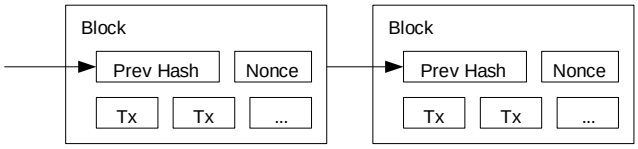
Solusi yang kami usulkan dimulai dengan server stempel waktu. Server stempel waktu bekerja dengan mengambil hash dari blok item yang akan diberi stempel waktu dan mempublikasikan hash tersebut secara luas, misalnya di surat kabar atau postingan Usenet [2-5]. Stempel waktu tersebut membuktikan bahwa data tersebut pasti sudah ada pada saat itu, tentu saja, agar dapat masuk ke dalam hash. Setiap stempel waktu menyertakan stempel waktu sebelumnya dalam hash-nya, membentuk rantai, dengan setiap stempel waktu tambahan memperkuat stempel waktu sebelumnya.



4. Bukti Kerja

Untuk mengimplementasikan server stempel waktu terdistribusi berbasis peer-to-peer, kita perlu menggunakan sistem proof-of-work yang mirip dengan Hashcash milik Adam Back [6], alih-alih postingan koran atau Usenet. Proof-of-work melibatkan pemindaian nilai yang ketika di-hash, seperti dengan SHA-256, hash-nya dimulai dengan sejumlah bit nol. Rata-rata pekerjaan yang diperlukan bersifat eksponensial dalam jumlah bit nol yang diperlukan dan dapat diverifikasi dengan mengeksekusi satu hash.

Untuk jaringan stempel waktu kita, kita mengimplementasikan proof-of-work dengan menambah satu nonce dalam blok hingga ditemukan nilai yang memberikan hash blok sejumlah bit nol yang diperlukan. Setelah upaya CPU dikeluarkan untuk membuatnya memenuhi proof-of-work, blok tersebut tidak dapat diubah tanpa mengulang pekerjaan tersebut. Karena blok-blok selanjutnya dirangkai setelahnya, pekerjaan untuk mengubah blok tersebut akan mencakup mengulang semua blok setelahnya.



Bukti kerja juga memecahkan masalah penentuan representasi dalam pengambilan keputusan mayoritas. Jika mayoritas didasarkan pada satu alamat IP satu suara, hal itu dapat dibantah oleh siapa pun yang mampu mengalokasikan banyak IP. Bukti kerja pada dasarnya adalah satu CPU satu suara. Keputusan mayoritas diwakili oleh rantai terpanjang, yang memiliki upaya bukti kerja terbesar yang diinvestasikan di dalamnya. Jika mayoritas daya CPU dikendalikan oleh node yang jujur, rantai yang jujur akan tumbuh paling cepat dan melampaui rantai pesaing mana pun. Untuk memodifikasi blok sebelumnya, penyerang harus mengulang bukti kerja blok dan semua blok setelahnya, lalu mengejar dan melampaui kinerja node yang jujur. Kami akan menunjukkan nanti bahwa probabilitas penyerang yang lebih lambat untuk mengejar berkurang secara eksponensial seiring dengan penambahan blok berikutnya.

Untuk mengimbangi peningkatan kecepatan perangkat keras dan minat yang bervariasi dalam menjalankan node dari waktu ke waktu, tingkat kesulitan bukti kerja ditentukan oleh rata-rata bergerak yang menargetkan jumlah rata-rata blok per jam. Jika dihasilkan terlalu cepat, tingkat kesulitannya meningkat.

5. Jaringan

Langkah-langkah untuk menjalankan jaringan adalah sebagai berikut:

1) Transaksi baru disiarkan ke semua node.
2) Setiap node mengumpulkan transaksi baru ke dalam satu blok.
3) Setiap node bekerja untuk menemukan bukti kerja yang sulit bagi bloknya.
4) Ketika sebuah node menemukan bukti kerja, ia menyiarkan blok tersebut ke semua node.
5) Node menerima blok hanya jika semua transaksi di dalamnya valid dan belum dibelanjakan.
6) Node menyatakan penerimaannya terhadap blok tersebut dengan berupaya menciptakan blok berikutnya dalam rantai, menggunakan hash dari blok yang diterima sebagai hash sebelumnya.

Node selalu menganggap rantai terpanjang sebagai rantai yang benar dan akan terus

memperpanjangnya. Jika dua node menyiarkan versi berbeda dari blok berikutnya secara bersamaan, beberapa node mungkin menerima salah satunya terlebih dahulu. Dalam hal ini, mereka mengerjakan blok pertama yang mereka terima, tetapi menyimpan cabang lainnya untuk berjaga-jaga jika cabang tersebut menjadi lebih panjang. Ikatan akan terputus ketika proof-of-work berikutnya ditemukan dan satu cabang menjadi lebih panjang; node yang bekerja di cabang lainnya kemudian akan beralih ke cabang yang lebih panjang.

Siaran transaksi baru tidak harus menjangkau semua node. Selama mencapai banyak node, transaksi tersebut akan segera masuk ke dalam blok. Siaran blok juga toleran terhadap pesan yang terputus. Jika sebuah node tidak menerima blok, node tersebut akan memintanya ketika menerima blok berikutnya dan menyadari bahwa ia melewatkan satu blok.

6. Insentif

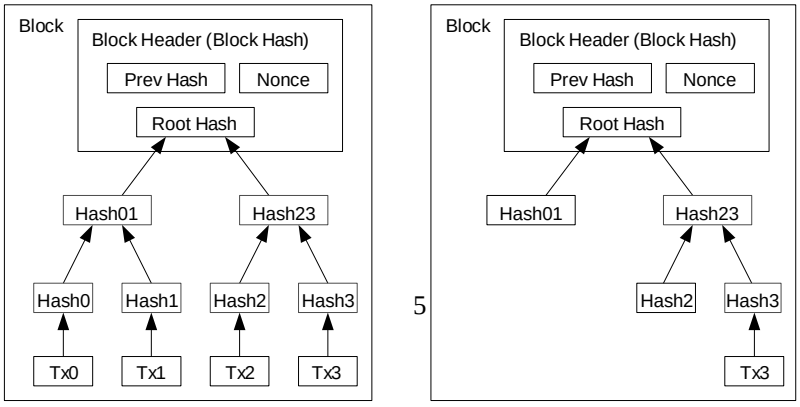
Berdasarkan konvensi, transaksi pertama dalam sebuah blok adalah transaksi khusus yang memulai koin baru milik pembuat blok tersebut. Hal ini menambah insentif bagi node untuk mendukung jaringan, dan menyediakan cara untuk mendistribusikan koin ke dalam sirkulasi, karena tidak ada otoritas pusat yang menerbitkannya. Penambahan koin baru secara bertahap dan konstan ini analog dengan penambang emas yang menghabiskan sumber daya untuk menambahkan emas ke dalam sirkulasi. Dalam kasus kami, yang dikeluarkan adalah waktu CPU dan listrik.

Insentif ini juga dapat didanai dengan biaya transaksi. Jika nilai keluaran suatu transaksi lebih kecil dari nilai masukannya, selisihnya adalah biaya transaksi yang ditambahkan ke nilai insentif blok yang berisi transaksi tersebut. Setelah sejumlah koin yang telah ditentukan memasuki sirkulasi, insentif dapat sepenuhnya beralih ke biaya transaksi dan sepenuhnya bebas inflasi.

Insentif ini dapat membantu mendorong node untuk tetap jujur. Jika seorang penyerang yang rakus mampu mengumpulkan daya CPU lebih besar daripada semua node yang jujur, ia harus memilih antara menggunakannya untuk menipu orang dengan mencuri kembali pembayarannya, atau menggunakannya untuk menghasilkan koin baru. Dia mesti merasa lebih menguntungkan untuk bermain sesuai aturan, aturan yang menguntungkannya dengan lebih banyak koin baru dibanding gabungan semua orang, daripada merusak sistem dan validitas kekayaannya sendiri.

7. Mendapatkan Kembali Ruang Disk

Setelah transaksi terbaru dalam sebuah koin terkubur di bawah blok yang cukup, transaksi yang telah digunakan sebelumnya dapat dibuang untuk menghemat ruang disk. Untuk memfasilitasi hal ini tanpa merusak hash blok, transaksi di-hash dalam Merkle Tree [7][2][5], dengan hanya root yang disertakan dalam hash blok. Blok lama kemudian dapat dipadatkan dengan memotong cabang-cabang pohon. Hash internal tidak perlu disimpan.

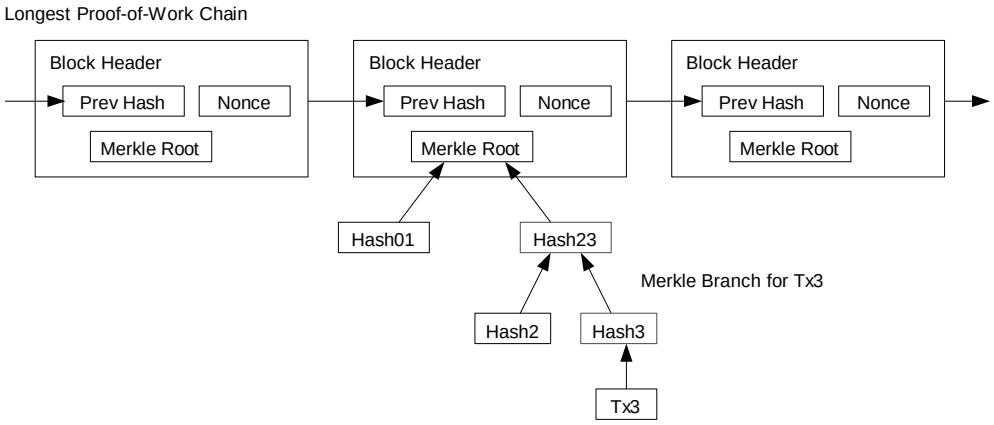


Transactions Hashed in a Merkle Tree After Pruning Tx0-2 from the Block

Header blok tanpa transaksi akan berukuran sekitar 80 byte. Jika kita asumsikan blok dihasilkan setiap 10 menit, 80 byte x 6 x 24 x 365 = 4,2 MB per tahun. Dengan sistem komputer yang biasanya dijual dengan RAM 2 GB pada tahun 2008, dan Hukum Moore yang memprediksi pertumbuhan saat ini sebesar 1,2 GB per tahun, penyimpanan seharusnya tidak menjadi masalah meskipun header blok harus disimpan di memori.

8. Simplified Payment Verification

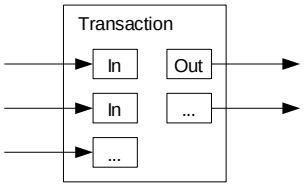
Verifikasi pembayaran dapat dilakukan tanpa menjalankan seluruh node jaringan. Pengguna hanya perlu menyimpan salinan header blok dari rantai bukti kerja terpanjang, yang dapat diperoleh dengan melakukan kueri pada node jaringan hingga yakin memiliki rantai terpanjang, dan mendapatkan cabang Merkle yang menghubungkan transaksi ke blok tempat transaksi tersebut diberi stempel waktu. Pengguna tidak dapat memeriksa transaksi itu sendiri, tetapi dengan menghubungkannya ke suatu tempat dalam rantai, ia dapat melihat bahwa node jaringan telah menerimanya, dan blok yang ditambahkan setelahnya akan semakin mengonfirmasi bahwa jaringan telah menerimanya.



Dengan demikian, verifikasi tersebut andal selama node yang jujur mengendalikan jaringan, tetapi lebih rentan jika jaringan dikuasai oleh penyerang. Meskipun node jaringan dapat memverifikasi transaksi sendiri, metode yang disederhanakan ini dapat dikelabui oleh transaksi rekayasa penyerang selama penyerang dapat terus menguasai jaringan. Salah satu strategi untuk melindungi diri dari hal ini adalah dengan menerima peringatan dari node jaringan ketika mereka mendeteksi blok yang tidak valid, yang akan mendorong perangkat lunak pengguna untuk mengunduh seluruh blok dan transaksi yang diperingatkan untuk mengonfirmasi ketidakkonsistenan tersebut. Bisnis yang sering menerima pembayaran kemungkinan besar tetap ingin menjalankan node mereka sendiri untuk keamanan yang lebih independen dan verifikasi yang lebih cepat.

9. Menggabungkan dan Membagi Nilai

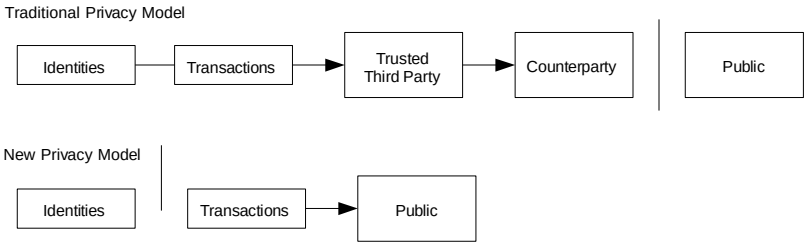
Meskipun koin dapat ditangani secara individual, akan sulit untuk melakukan transaksi terpisah untuk setiap sen dalam transfer. Agar nilai dapat dibagi dan digabungkan, transaksi mengandung beberapa masukan dan keluaran. Biasanya akan ada satu masukan dari transaksi sebelumnya yang lebih besar atau beberapa masukan yang menggabungkan jumlah yang lebih kecil, dan maksimal dua keluaran: satu untuk pembayaran, dan satu lagi untuk mengembalikan kembalian, jika ada, kepada pengirim.



Perlu dicatat bahwa fan-out, di mana suatu transaksi bergantung pada beberapa transaksi, dan transaksi-transaksi tersebut bergantung pada lebih banyak transaksi lainnya, bukanlah masalah di sini. Tidak perlu mengekstrak salinan lengkap riwayat transaksi secara mandiri.

10. Privasi

Model perbankan tradisional mencapai tingkat privasi dengan membatasi akses informasi hanya kepada pihak-pihak yang terlibat dan pihak ketiga yang tepercaya. Keharusan untuk mengumumkan semua transaksi secara publik menghalangi metode ini, tetapi privasi tetap dapat dipertahankan dengan memutus aliran informasi di tempat lain: dengan menjaga kerahasiaan kunci publik. Publik dapat melihat bahwa seseorang mengirimkan sejumlah uang kepada orang lain, tetapi tanpa informasi yang menghubungkan transaksi tersebut dengan siapa pun. Hal ini serupa dengan tingkat informasi yang dirilis oleh bursa saham, di mana waktu dan ukuran transaksi individual, "rekaman", dipublikasikan, tetapi tanpa mengungkapkan siapa pihak-pihak tersebut.



Sebagai firewall tambahan, pasangan kunci baru harus digunakan untuk setiap transaksi agar tidak terhubung ke pemilik yang sama. Beberapa penautan masih tidak dapat dihindari pada transaksi multi-input, yang tentu saja akan mengungkapkan bahwa input-input tersebut dimiliki oleh pemilik yang sama. Risikonya adalah jika pemilik kunci terungkap, penautan tersebut dapat mengungkapkan transaksi lain yang dimiliki oleh pemilik yang sama.

11. Perhitungan

Kami mempertimbangkan skenario penyerang yang mencoba menghasilkan rantai alternatif lebih cepat daripada rantai jujur. Sekalipun hal ini tercapai, sistem tidak akan terbuka terhadap perubahan sewenang-wenang, seperti menciptakan nilai secara tiba-tiba atau mengambil uang yang bukan milik penyerang. Node tidak akan menerima transaksi yang tidak valid sebagai pembayaran, dan node jujur tidak akan pernah menerima blok yang berisi transaksi tersebut. Penyerang hanya dapat mencoba mengubah salah satu transaksinya sendiri untuk mengambil kembali uang yang baru saja dibelanjakannya.

Perlombaan antara rantai jujur dan rantai penyerang dapat dikarakterisasikan sebagai Binomial Random Walk. Peristiwa suksesnya adalah rantai jujur yang diperpanjang satu blok, meningkatkan keunggulannya sebesar +1, dan peristiwa gagalnya adalah rantai penyerang yang diperpanjang satu blok, mengurangi selisihnya sebesar -1.

Probabilitas penyerang mengejar ketertinggalan dari defisit yang diberikan analog dengan masalah Kehancuran Penjudi. Misalkan seorang penjudi dengan kredit tak terbatas memulai dengan defisit dan memainkan percobaan yang berpotensi tak terbatas jumlahnya untuk mencoba mencapai titik impas. Kita dapat menghitung kemungkinan dia mencapai titik impas, atau bahwa penyerang dapat mengejar rantai yang jujur, sebagai berikut [8]:

p = kemungkinan node jujur menemukan blok berikutnya
q = kemungkinan penyerang menemukan blok berikutnya
q_z = kemungkinan penyerang akan mengejar dari z blok di belakang

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

Dengan asumsi kita bahwa p > q, probabilitasnya turun secara eksponensial seiring bertambahnya jumlah blok yang harus dikejar penyerang. Dengan peluang yang tidak menguntungkannya, jika ia tidak melakukan serangan mendadak yang beruntung sejak awal, peluangnya menjadi sangat kecil karena ia semakin tertinggal.

Sekarang kita pertimbangkan berapa lama penerima transaksi baru perlu menunggu sebelum cukup yakin bahwa pengirim tidak dapat mengubah transaksi. Kita asumsikan pengirim adalah penyerang yang ingin membuat penerima percaya bahwa ia telah membayarnya untuk sementara waktu, lalu mengalihkannya untuk membayar kembali kepada dirinya sendiri setelah beberapa waktu berlalu. Penerima akan diberitahu ketika hal itu terjadi, tetapi pengirim berharap sudah terlambat.

Penerima menghasilkan pasangan kunci baru dan memberikan kunci publik kepada pengirim sesaat sebelum penandatanganan. Hal ini mencegah pengirim mempersiapkan rantai blok sebelumnya dengan mengerjakannya terus menerus hingga ia cukup beruntung untuk maju cukup jauh, lalu mengeksekusi transaksi pada saat itu juga. Setelah transaksi dikirim, pengirim yang tidak jujur mulai bekerja secara rahasia pada rantai paralel yang berisi versi alternatif dari transaksinya. Penerima menunggu hingga transaksi ditambahkan ke sebuah blok dan z blok telah ditautkan setelahnya. Ia tidak tahu persis berapa banyak kemajuan yang telah dicapai penyerang, tetapi dengan asumsi blok yang jujur membutuhkan waktu rata-rata yang diharapkan per blok, potensi kemajuan penyerang akan berdistribusi Poisson dengan nilai yang diharapkan:

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

Untuk mendapatkan probabilitas penyerang masih bisa mengejar sekarang, kita kalikan kerapatan Poisson untuk setiap jumlah kemajuan yang bisa dicapainya dengan probabilitas ia bisa mengejar dari titik tersebut:

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

Penataan ulang untuk menghindari penjumlahan ekor distribusi yang tak terhingga...

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

Mengonversi ke kode C...

```
#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;
}
```

Dengan menjalankan beberapa hasil, kita dapat melihat probabilitas menurun secara eksponensial dengan 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
```

Memecahkan P kurang dari 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. Kesimpulan

Kami telah mengusulkan sebuah sistem untuk transaksi elektronik tanpa bergantung pada kepercayaan. Kami memulai dengan kerangka kerja koin yang umum, yang terbuat dari tanda tangan digital, yang memberikan kontrol kepemilikan yang kuat, tetapi tidak lengkap tanpa cara untuk mencegah pengeluaran ganda. Untuk mengatasi hal ini, kami mengusulkan jaringan peer-to-peer menggunakan proof-of-work untuk merekam riwayat transaksi publik yang dengan cepat menjadi tidak praktis secara komputasi bagi penyerang untuk diubah jika node yang jujur mengendalikan sebagian besar daya CPU. Jaringan ini tangguh dalam kesederhanaannya yang tidak terstruktur. Node bekerja sekaligus dengan sedikit koordinasi. Mereka tidak perlu diidentifikasi, karena pesan tidak diarahkan ke tempat tertentu dan hanya perlu dikirimkan berdasarkan upaya terbaik. Node dapat meninggalkan dan bergabung kembali dengan jaringan sesuka hati, menerima rantai proof-of-work sebagai bukti atas apa yang terjadi selama mereka pergi. Mereka memberikan suara dengan kekuatan CPU mereka, menyatakan penerimaan mereka terhadap blok yang valid dengan berupaya memperluasnya dan menolak blok yang tidak valid dengan menolak untuk mengerjakannya. Aturan dan insentif apa pun yang diperlukan dapat ditegakkan dengan mekanisme konsensus ini.

Referensi

[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.