بیت‌کوین: یک سیستم پول نقد الکترونیکی همتابه‌همتا

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

Translated in Persian from bitcoin.org/bitcoin.pdf
by Zahra Amini

چکیده. یک نسخه‌ی سددرسد همتابه‌همتا از پول نقد الکترونیکی، فرستاده‌شدن سرراست پرداخت‌های آنلاین را از یک فرد به فرد دیگر بدون نیاز به گذر از نهاد مالی امکان‌پذیر می‌کند. امضای دیجیتال بخشی از راه‌حل است، ولی اگر همچنان برای پیشگیری از بازخرج به وجود یک میانجی نیاز باشد، برتری‌های اصلی آن از بین می‌رود. ما، با استفاده از یک شبکه‌ی همتابه‌همتا، راهکاری برای مشکل خرج دوباره‌ی پول به میان می‌آوریم. این شبکه، با درهم‌سازی[۱] تراکنش‌ها در چهارچوب یک زنجیره‌ی پیوسته بر پایه‌ی اثباتِ انجام کارِ بر پایه‌ی درهم، برچسب زمان‌دارشان می‌کند؛ پیشینه‌ی بدست آمده بدون انجام دوباره‌ی کاری که پیش‌تر انجام‌شده، ویرایش‌پذیر نخواهد بود. بلندترین زنجیره نه‌تنها اثبات چیدمان پیش‌آمدهاست، که اثبات پیدایش خود از بزرگترین گردآورد توان پردازشی‌هم است. تا زمانی که گره‌هایی[۲] که در حمله به شبکه همکاری نمی‌کنند، توان پردازشی بیشتر در شبکه را در دست داشته باشند، بلندترین زنجیره را ساخته و از مهاجمان پیشی می‌گیرند. خود شبکه به کمترین ساختار نیاز دارد. پیام‌ها بر پایه‌ی بهترین تلاش فرستاده میشوند، و گره‌ها، با پذیرش بلندترین زنجیره‌یِ اثباتِ کار به عنوان چیزی که در نبود آنها رخ داده، می‌توانند آزادانه شبکه را ترک کنند یا به آن بپیوندند.

۱- پیش‌گفتار

امروزه تجارت در اینترنت کمابیش به‌شکل ویژه به نهادهای مالی، به عنوان میانجی برای پردازش پرداخت‌های الکترونیکی، تکیه کرده است. اگرچه این سامانه به اندازه کافی پاسخگوی بیشتر تراکنش‌هاست، ولی ضعف‌های بنیادیِ روش‌های بر پایه‌ی اعتماد هم دارد. انجام تراکنش‌های قطعی و بازگشت‌ناپذیر به‌راستی ممکن نیست؛ چراکه نهادهای مالی نمی‌توانند از میانجی‌گری سر باز بزنند. این میانجی‌گری هزینه‌های تراکنش را افزایش می‌دهد، کمترین اندازه‌ی قابل استفاده‌ی تراکنش را محدود و انجام تراکنش‌های کوچک مرسوم را ناممکن می‌کند. ناممکن بودن انجام تراکنش‌های بازگشت‌ناپذیر در ازای خدمات بازگشت‌ناپذیر بهای سنگین‌تری‌ست که پرداخت می‌شود چراکه

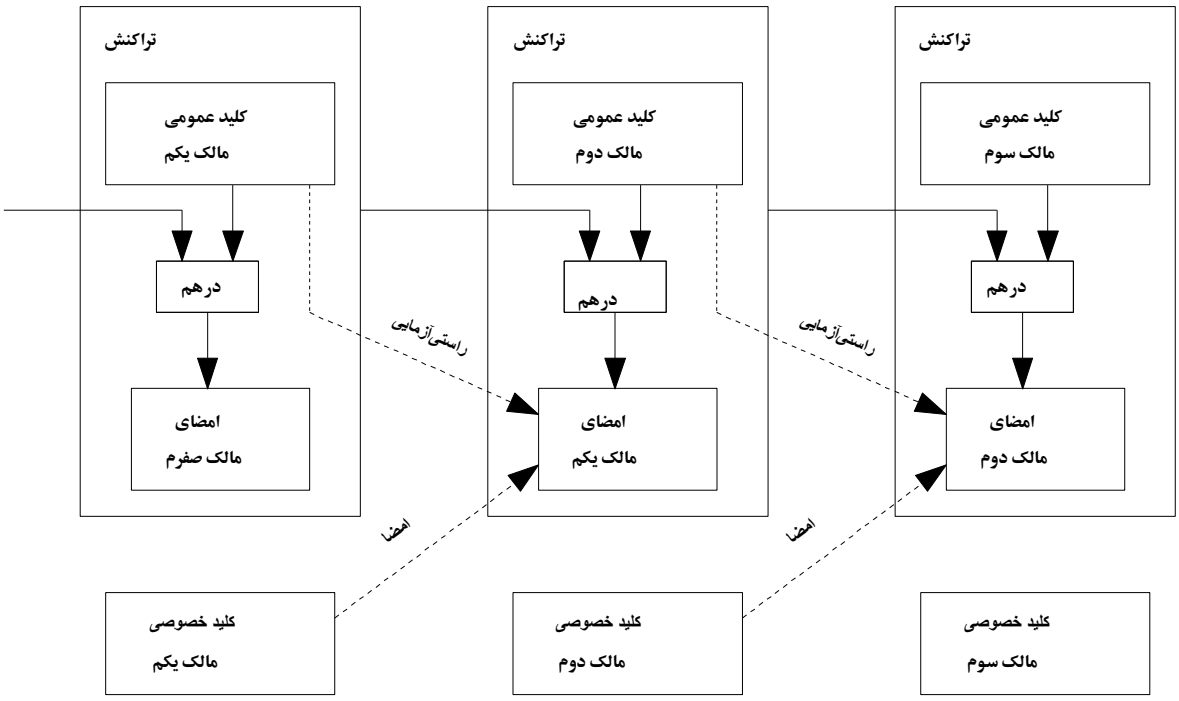
۱ hash(v)
۲ node

با امکان بازگشت، نیاز به اعتماد افزایش می‌یابد. فروشندگان باید نسبت به مشتریان خود محتاطانه عمل کرده و با درخواست اطلاعات بی‌مورد آن‌ها را آزار دهند. همواره درسد مشخصی کلاهبرداری، گریزناپذیر در نظر گرفته شده است. این هزینه‌ها و شبهه‌های پرداخت در پرداخت‌های حضوری با استفاده از ارزهای فیزیکی پیشگیری‌پذیر است؛ ولی هم‌اکنون سازوکاری برای پرداخت از راه ارتباطی، بدون میانجی، وجود ندارد.

بجای اعتماد، یک سامانه‌ی پرداخت الکترونیکی بر پایه‌ی رمزنگاری نیاز است تا هر دو طرف بدون نیاز به میانجی بتوانند سرراست تراکنش داشته باشند. تراکنش‌هایی که از لحاظ محاسباتی بازگشت‌ناپذیراند از فروشنده در برابر کلاهبرداری محافظت می‌کنند؛ افزون بر این، سازوکارِ تضمینِ دریافت می‌تواند به راحتی پیاده‌سازی شده تا خریدار را نیز در امان نگه دارد. در این نوشتار ما راهکاری برای مشکل بازخرج[1] با استفاده از یک سرورِ برچسب‌زمانِ توزیع‌شده‌ی همتابه‌همتا، پیشنهاد می‌کنیم که اثباتی محاسباتی از چیدمانِ زمانِ قرارگیریِ تراکنش‌ها می‌سازد. این سامانه تا زمانی که گره‌های درست‌کار، روی هم رفته توان پردازشی بیشتری از گره‌های مهاجم در دست دارند، ایمن است.

۲- تراکنش

ما یک سکه‌ی الکترونیکی را با زنجیره‌ای از امضاهای دیجیتالی تعریف می‌کنیم. هر مالک، برای جابجایی سکه به فرد دیگر، درهم تراکنش پیشین و کلید عمومی مالک بعدی را امضا کرده و به انتهای سکه ضمیمه می‌کند. دریافت‌کننده می‌تواند از راه راستی‌آزماییِ این امضاها زنجیره‌ی مالکیت را نیز راستی‌آزمایی کند.

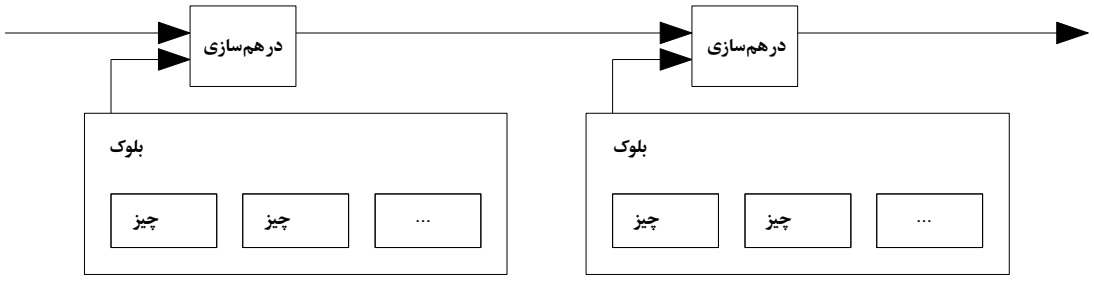


مشکل اینجاست که دریافت‌کننده نمی‌تواند بازخرج نشدن سکه توسط مالکان پیشین را راستی‌آزمایی کند. نخستین راهکاری که به ذهن می‌رسد، این است که ضراب‌خانه متمرکز معتمدی در نظر بگیریم تا تک تک تراکنش‌ها را بررسی کند. پس از هر تراکنش، سکه به ضراب‌خانه بازگردانده می‌شود تا در ازای آن یک سکه‌ی تازه زده شود؛ تنها سکه‌هایی که سرراست از ضراب‌خانه جابجا داده شده‌اند بازخرج نشده شمرده می‌شوند. مشکل این راه‌حل این است که با اجبار گذر تراکنش‌ها از ضراب‌خانه، سرنوشت کل سامانه‌ی پول به گروهی که ضراب‌خانه را اداره می‌کنند وابسته است؛ درست مانند سامانه‌ی بانکی.

ما روشی نیاز داریم که دریافت‌کننده بتواند از پیش‌تر امضا نشدن تراکنش(ی که این سکه را خرج می‌کند) توسط مالکان پیشین مطمئن شود. برای دست‌یابی به این هدف، نخستین تراکنش را تراکنش درست می‌شماریم و تلاش‌های بعدی برای بازخرج را نادیده می‌گیریم. آگاهی از تمام تراکنش‌ها، تنها راه بررسی نبود یک تراکنش است. در روش برپایه‌ی ضراب‌خانه، این مرکز با آگاهی از تمامی تراکنش‌ها، چیدمان آن‌ها را روشن می‌کرد. برای بدست‌آوردن این نتیجه، بدون نیاز به میانجی، باید تراکنش‌ها به صورت عمومی فرستاده شوند [1] همچنین به سامانه‌ای نیاز است که کاربران آن بر سر یک تاریخچه‌ی یکسان از چیدمانی که تراکنش‌ها دریافت شده‌است، توافق کنند. از طرفی دریافت‌کننده به اثباتی نیاز دارد که نشان دهد در زمان تراکنش، بیشتر گره‌ها بر سر دریافت آن برای نخستین بار اتفاق نظر داشته‌اند.

۳- سرور برچسبِ زمان

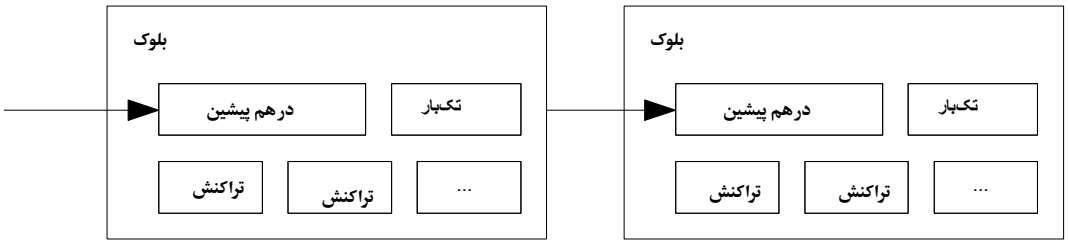
راه‌حل پیشنهادی ما با سرور برچسبِ زمان آغاز می‌شود. بطور کلی سرور برچسب زمان درهم بلوکی از چیزهایی که قرار است برچسب‌زمان‌دار شوند را ساخته سپس درهم را به طور گسترده منتشر می‌کند؛ همانند روزنامه و یا به شکل پستی در یوزنت [2-5]. بدیهی است که برای محاسبه درهم نیاز به داده(ی درهم‌سازی شده) است، پس وجود این برچسبِ زمان در هر لحظه وجود آن چیزها در بلوک در زمان نشر برچسب را اثبات می‌کند. هر برچسب زمان، برچسب زمانِ پیش از خود را در درهم خود در بر می‌گیرد، در مجموع یک زنجیره را شکل می‌دهند و هر برچسب زمان تازه، برچسب زمان‌های پیش از خود را تقویت می‌کند.



۴- اثبات انجام کار

برای پیاده‌سازی سرور برچسبِ زمانِ توزیع‌شده‌ی همتابه‌همتا، به‌جای بکار بردن یادداشت‌های روزنامه و یا یوزنت، نیاز به سامانه‌ی اثباتِ انجام کاری مانند سامانه‌ی هشکش آدام بک داریم[6]. سامانه‌ی اثباتِ انجامِ کار شامل وارسی برای مقداری است که اگر درهم[۳] آن (برای مثال SHA-256) محاسبه شَوَد، با تعدادی بیت (با مقدار) صفر آغاز می‌شود. میانگین کار مورد نیاز، نمایی و از توان تعداد بیت‌های مقدار-صفر خواسته شده است و با انجام تنها یک درهم‌سازی قابل راستی‌آزمایی است.

پیاده‌سازی اثبات انجام کار برای شبکه‌ی برچسبِ زمان‌مان این گونه است که تک‌بار[۴] در بلوک را افزایش می‌دهیم تا مقداری یافت شود که به درهم بلوک، بیت‌های مقدار-صفر مورد نیازش را بدهد. زمانی که تلاش پردازشگر مقدار مورد نظرِ اثباتِ انجامِ کار را برآورده سازد، بلوک، دیگر بدون انجام دوباره کاری که انجام‌شده، ویرایش‌پذیر نخواهد بود. همین‌طور که بلوک‌های بعدی به آن زنجیر می‌شوند، برای ویرایش بلوک باید کار انجام‌شده روی تمامی بلوک‌های بعدی هم دوباره انجام گیرد.



همچنین سامانه‌ی اثبات انجام کار، چاره‌ای برای شناخت دیدگاه چیره در تصمیم‌گیری‌های جمعی است. اگر اکثریت بر پایه‌ی هر آدرسِ آی‌پی، یک رای بود، کسی که می‌توانست به آی‌پی‌های زیادی دسترسی داشته باشد، می‌توانست این تصمیم‌گیری را بهم بزند. در سامانه‌ی اثبات انجام کار هر پردازشگر در نهایت یک رای دارد. تصمیم اکثریت با بلندترین زنجیره مشخص می‌شود که بیشترین اثبات انجام کار در آن هزینه شده است. چنانچه بیشتر توان پردازشی در دست گره‌های درست‌کار باشد، زنجیره‌ی درست زودتر رشد کرده و از زنجیره‌های رقیب پیشی می‌گیرد. برای ویرایش یک بلوک پیشین، فرد مهاجم باید علاوه بر برآورده‌سازی دوباره‌ی اثبات انجام کار آن بلوک، اثبات انجام‌کار تمامی بلوک‌های پس از آن را نیز دوباره برآورده سازد و سپس به گره‌های درست‌کار رسیده و از آن‌ها پیشی بگیرد. در ادامه نشان خواهیم داد که احتمال اینکه یک مهاجم به این نقطه برسد با اضافه شدن بلوک‌های بعدی به‌صورت نمایی کاهش می‌یابد.

۳ hash
۴ nonce: number used once

برای جبران افزایش سرعت سخت‌افزار و دگرگونی تعداد گره‌های فعال در بازه‌های زمانی گوناگون، سختی اثبات انجام کار با میانگینِ‌متحرک‌گیری از بلوک‌های ساخته شده بر ساعت مشخص می‌شود. اگر بلوک‌ها با سرعت زیادی ساخته شوند، سختی افزایش می‌یابد.

۵- شبکه

شبکه را می‌توان با گام‌های زیر اجرا کرد:

۱) تراکنش‌های تازه به تمامی گره‌ها فرستاده می‌شوند.
۲) هر گره تراکنش‌های تازه را در یک بلوک گرد می‌آورد.
۳) هر گره تلاش می‌کند تا یک اثباتِ انجامِ کارِ به اندازه‌یِ کافی دشوار را برای بلوک خود بیابد.
۴) زمانی که یک گره اثبات انجام کار را می‌یابد، بلوک خود را به تمامی گره‌ها می‌فرستد.
۵) دیگر گره‌ها آن بلوک را تنها در شرایطی می‌پذیرند که همه‌ی تراکنش‌های آن درست باشند و پیش‌تر خرج نشده باشند.
۶) دیگر گره‌ها موافقت خود با بلوک مورد نظر را با ساختن بلوک بعدی در زنجیره اعلام می‌کنند و به این منظور از درهم بلوک پذیرفته‌شده به عنوان درهم پیشین استفاده می‌کنند.

گره‌ها همواره بلندترین زنجیره را به عنوان زنجیره‌ی درست می‌شناسند و بر روی بلندتر کردن آن کار می‌کنند. اگر دو گره دو نسخه‌ی گوناگون از بلوک بعدی را به صورت همزمان پخش کنند، ممکن است برخی گره‌ها یک نسخه و برخی گره‌ها نسخه‌ی دیگر را دریافت کنند. در این شرایط آنها بر روی نخستین نسخه‌ای که دریافت کرده‌اند کار می‌کنند؛ ولی شاخه دیگر را ذخیره می‌کنند تا اگر بلندتر شد، آن را ادامه دهند. این اتصال زمانی قطع می‌شود که اثباتِ انجامِ کارِ پیدا شده و یکی از دو زنجیره از دیگری بلندتر شود. گره‌هایی که روی شاخه دیگر کار می‌کردند به زنجیره‌ی بلندتر تغییر مسیر می‌دهند.
نیازی نیست که تراکنش‌های تازه منتشرشده به همه‌ی گره‌ها برسند. همین که به تعدادی از گره‌ها برسد، یعنی به زودی در یک بلوک قرار خواهند گرفت. همچنین بلوک‌های منتشرشده نسبت به پیام‌های از قلم افتاده مقاوم‌اند؛ به این معنی که اگر یک گره بلوکی را دریافت نکند، با دریافت بلوک بعدی، متوجه نبود بلوک خواهد شد و آن را نیز درخواست می‌کند.

۶- انگیزه

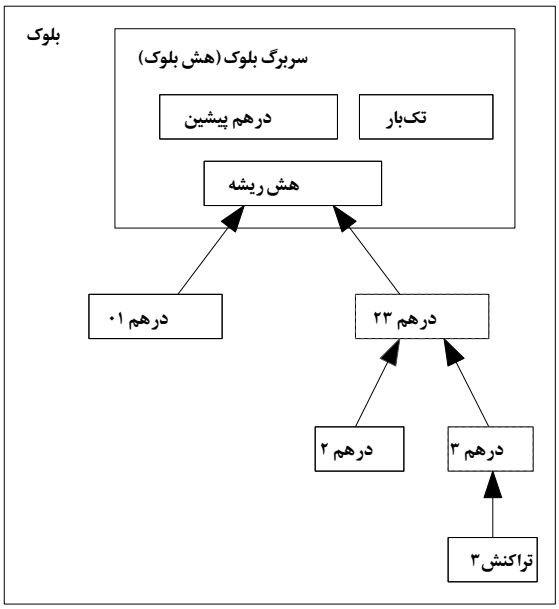
به‌صورت قراردادی، نخستین تراکنش در هر بلوک، تراکنش ویژه‌ای شمرده می‌شود که سکه ای تازه را ساخته و مالک آن سازنده‌ی آن بلوک است. این مساله انگیزه‌ای برای گره‌ها می‌سازد که از شبکه پشتیبانی کنند و روشی برای توزیع آغازین سکه‌ها برای به گردش درآوردن آن‌ها مهیا کنند؛ چراکه هیچ مرکزیتی برای توزیع آن‌ها موجود نیست. می‌توانیم این رویه‌ی پیاپی ساخت مقدار ثابتی از سکه‌های تازه را به استخراج‌کنندگان طلا تشبیه کنیم که با هزینه‌کردن منابع، طلای تازه را به گردش درمی‌آورند. در مقوله‌ی ما، این منابع زمان پردازشگر و برق هزینه‌شده است.

انگیزه همچنین می‌تواند با کارمزد تراکنش‌ها نیز تامین گردد. اگر مقدار خروجی یک تراکنش از مقدار ورودی آن کمتر باشد، مابه‌تفاوت کارمزد تراکنش است که به میزان انگیزه بلوک حاوی تراکنش می‌افزاید. زمانی که تعداد از پیش نشان شده‌ی سکه‌ها به گردش درآمد، انگیزه تنها می‌تواند به کارمزد تراکنش‌ها خلاصه شود و به راستی عاری از تورم باشد.

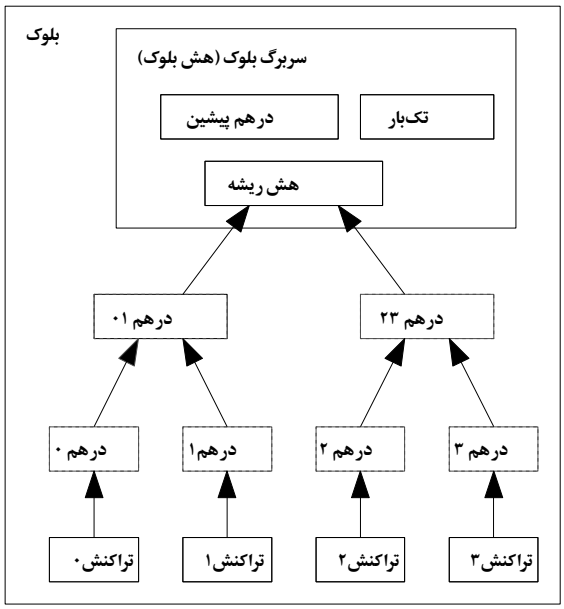
این انگیزه می‌تواند گره‌ها را تشویق به درست‌کار ماندن کند. اگر یک مهاجم آزمند بتواند توان پردازشی بیشتری از تمامی گره‌های درست‌کار جمع کند، باید بین استفاده‌ی این توان برای سرقت پرداخت‌هایی که خودش به مردم داشته و کلاهبرداری از آن‌ها و یا برای ساخت سکه‌های تازه، انتخاب کند. برای این شخص پیروی از قانون باید سودمندتر باشد؛ قانون‌هایی که به او از دیگران روی هم رفته، سکه‌های تازه‌ی بیشتری بدهد تا او نخواهد سامانه و ارزش سرمایه‌ی خود را پایمال کند.

۷- بازیابی فضای دیسک

به محض اینکه آخرین تراکنش در یک سکه زیر بلوک‌های کافی قرار گرفت، تراکنش‌های خرج‌شده‌ی پیش از آن می‌توانند نادیده گرفته شوند تا فضای دیسک محفوظ باشد. برای انجام این کار بدون نیاز به شکستن درهم بلوک، تراکنش‌ها در یک درخت مرکل درهم‌سازی می‌شوند [7] [2] [5] که تنها ریشه‌ی این درخت مرکل در درهم بلوک قرار می‌گیرد. بنابراین بلوک‌های کهنه به این روش می‌توانند با استفاده از هرس شاخه‌های درخت فشرده شوند. نیازی به ذخیره‌ی درهم‌های داخلی نیست.



پس از هرس تراکنش‌های ۰ تا ۲ از بلوک



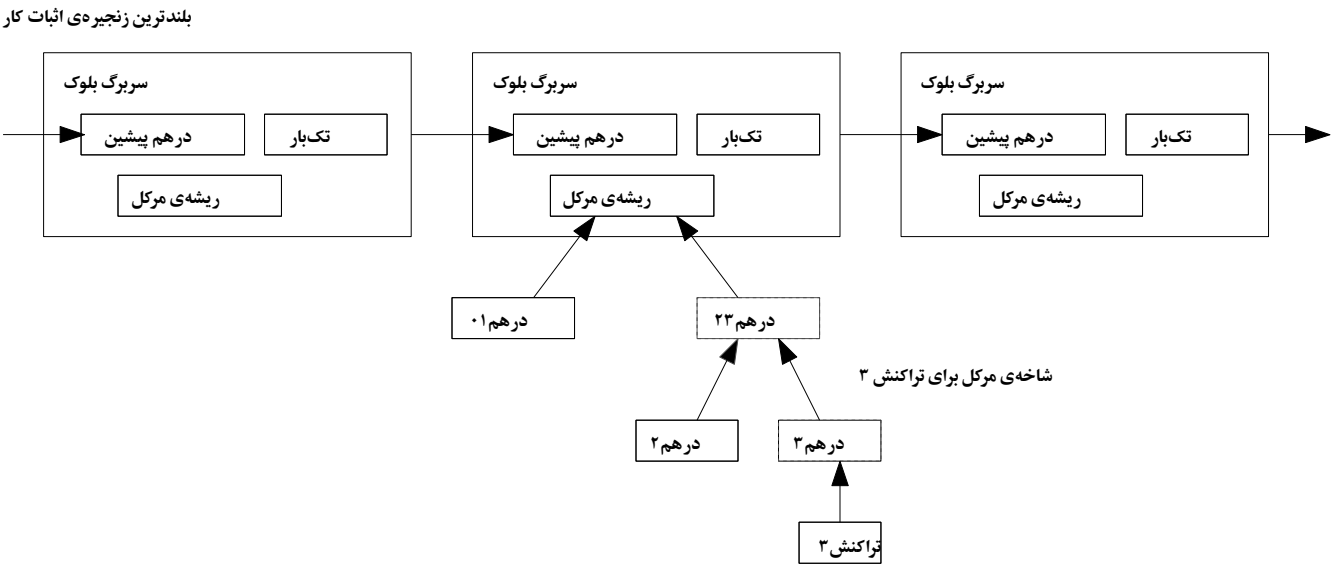
تراکنش‌های هش شده در یک درخت مرکل

حجم سربرگ بلوکی که حاوی تراکنش نیست کم و بیش حدود ۸۰ بایت خواهد بود. چنانچه زمان ساخت بلوک‌ها را هر ۱۰ دقیقه بگیریم، ۸۰ بایت* ۶*۲۴*۳۶۵= ۴٫۲ مگابایت بر سال خواهد بود. سامانه‌های کامپیوتری که در سال ۲۰۰۸ به فروش می‌رسند بیشتر دارای رم ۲ گیگابایتی‌اند و قانون مور رشد کنونی را سالانه ۱.۲ گیگابایت برآورد می‌کند؛ فضای ذخیره‌سازی هتا در صورت نیاز به نگهداریِ سربرگِ بلوک‌ها در حافظه، نباید مشکل‌ساز باشد.

۸-راستی‌آزمایی ساده‌شده‌ی پرداخت

راستی‌آزمایی پرداخت‌ها بدون اجرای گره کامل شبکه[۵] شدنی است. کاربر تنها باید رونوشتی از سربرگ بلوک‌های بلندترین زنجیره‌ی اثباتِ کار، که می‌تواند با پرسمان از گره‌های شبکه تا زمانی که اطمینان حاصل کند بلندترین زنجیره را دارد بدست آید، را نگه دارد و شاخه‌ی مرکلی که تراکنش را به آن بلوک، که به آن برچسب‌زمانی شده است، پیوند می‌دهد را بدست آورد. او نمی‌تواند به تنهایی تراکنش بررسی کند، ولی با پیوند آن به مکانی در زنجیره، می‌تواند پذیرفته شدن آن توسط یک گره شبکه را ببیند و بلوک‌های افزوده‌شده‌ی پس از آن گواهی بر این است که شبکه آن را پذیرفته است.

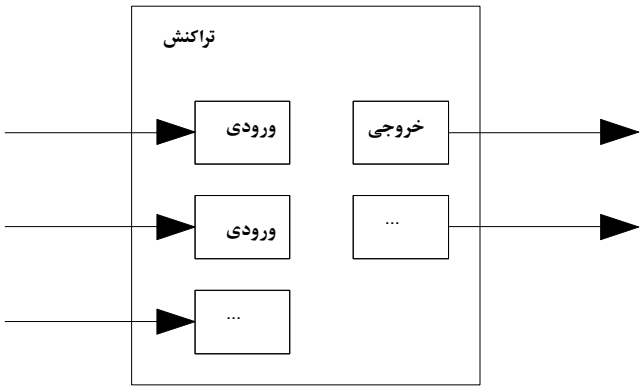
۵ a full network node



به این گونه تا زمانی که گره‌های درست‌کار شبکه را در دست دارند، این راستی‌آزمایی قابل اعتماد و اگر مهاجم بر شبکه چیره شده باشد، آسیب‌پذیرتر است. در حالی که گره‌های شبکه می‌توانند به تنهایی تراکنش‌ها را راستی‌آزمایی کنند، تا زمانی که مهاجم بر شبکه بتواند چیره بماند، روش ساده‌شده با تراکنش‌های دروغین مهاجم قابل فریفتن است. یک راهبرد برای پیشگیری، این است که گره‌های شبکه هنگام شناسایی بلوکی نامعتبر به نرم‌افزار کاربر هشدار داده تا نرم‌افزار کاربر همه‌ی بلوک و تراکنش‌های مشکوک را بارگیری و ناهماهنگی را تایید کند. کسب‌وکارهایی که پرداخت‌های پی‌درپی دارند بهتر است گره‌های خود را اجرا کنند تا ایمنی جداگانه‌ی بیشتری و راستی‌آزمایی زودتری داشته باشند.

۹- جداسازی و ترکیب مقدار

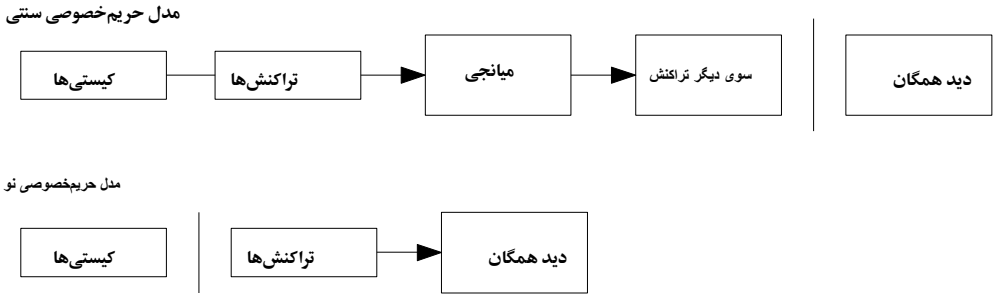
اگرچه مدیریت سکه‌ها به صورت جداگانه ممکن است؛ ولی ساختن یک تراکنش جدا برای جابجایی هر سِنت ناکارآمد به نظر می‌رسد. برای اینکه جداسازی و ترکیب هر مقدار ممکن باشد، تراکنش‌ها دارای چندین ورودی و خروجی‌اند. به طور معمول یا یک ورودی، از یک تراکنشِ پیشین بزرگتر، یا چندین ورودی، که مقدارهای خردتر را ترکیب می‌کنند، و بیشینه دو خروجی خواهیم داشت: یکی برای پرداخت و دیگری برای باقی‌مانده‌ای، که اگر باشد، به فرستنده باز می‌گردد.



باید بازگفت که پخش شدن ورودی تراکنش، که تراکنش به تراکنش‌های پرشماری و آن تراکنش‌ها به تراکنش‌های بیشتری وابسته باشند، در این جا مشکل‌ساز نیست. هیچ‌گاه نیازی به استخراج جداگانه‌ی تاریخچه‌ی کامل یک تراکنش نخواهد بود.

۱۰- حریم خصوصی

روش بانک‌داری سنتی با محدود کردن دسترسی به اطلاعات به طرف‌های تراکنش و میانجی به ترازی از حریم خصوصی دست می‌یابد. نیاز به فرستادن همه‌ی تراکنش‌ها در دید همگان این روش را ناممکن می‌کند؛ ولی حریم خصوصی همچنان می‌تواند با شکستن جریان اطلاعات در جایی دیگر، حفظ شود: با ناشناس نگه‌داشتن کلیدهای عمومی. همه می‌توانند فرستاده‌شدن مقداری از یک فرد به فرد دیگر را ببینند ولی بدون اطلاعاتی که تراکنش را به شخصی خاص مرتبط کند. همانند اطلاعات منتشر شده از تالار بورس است که زمان و اندازه‌ی هر خرید و فروش، «نوار»، بدون افشای هویت افراد، در دید همگان است.



به عنوان یک لایه‌ی محافظتی دیگر، برای هر تراکنش بایستی یک جفت کلید تازه به کار رود تا نتوان آن‌ها را به صاحبی یکسان ربط داد. البته مقداری ربط‌پذیری با تراکنش‌های دارای چند ورودی گریزناپذیر است؛ چراکه روشن است ورودی آن‌ها سددرسد از یک مالک بوده است. خطر این مساله این است که اگر صاحب کلید مشخص شود، ارتباط او با دیگر تراکنش‌هایش نمایان خواهد شد.

۱۱- محاسبات

ما سناریویی را در نظر میگیریم که در آن مهاجم تلاش می‌کند زنجیره‌ی جایگزینی را تندتر از زنجیره‌ی درست بسازد. هتا اگر موفق به این کار شود، سامانه به ویرایش‌های دلبخواهی مانند ساخت ارزش از هیچ و یا دریافت پولی که زمانی در دست مهاجم نبوده است، آسیب‌پذیر نخواهد بود. گره‌ها تراکنش‌های نامعتبر را به عنوان پرداخت نمی‌پذیرند و گره‌های درست‌کار هرگز بلوک‌های حاوی این تراکنش‌ها را نخواهند پذیرفت. یک مهاجم تنها می‌تواند برای ویرایش یکی از تراکنش‌های خود تلاش کند تا پولی که به تازگی خرج کرده است را پس بگیرد.

رقابت بین زنجیره‌ی درست و زنجیره‌ی مهاجم را میتوان به عنوان گشت تصادفی دوجمله‌ای توصیف کرد. پیروزی، بلندتر شدن زنجیره‌ی درست به اندازه‌ی یک بلوک است، با افزایش +1 پیشی و شکست، بلندتر شدن زنجیره‌ی مهاجم به اندازه‌ی یک بلوک است، با کاهش -1 اختلاف.

احتمال اینکه مهاجم بتواند با یک عقب افتادگی فرض شده به زنجیره‌ی درست برسد، همانند مساله پاکباختگی قمارباز است. تصور کنید یک قمارباز با موجودی نامحدود با یک عقب افتادگی بازی را شروع کند و با تعداد بالقوه نامحدودی شانسش را برای تلاش به رسیدن به نقطه سربه‌سر بازی می‌کند. ما می‌توانیم احتمال اینکه به نقطه سربه‌سر برسد و یا اینکه یک مهاجم به زنجیره‌ی درست برسد را اینگونه محاسبه کنیم[8]:

p = احتمال اینکه یک گره درست‌کار بلوک بعدی را پیدا کند

q = احتمال اینکه مهاجم بلوک بعدی را پیدا کند

qz = احتمال اینکه مهاجم از z بلوک پشت‌تر بتواند به زنجیره‌ی درست برسد

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

با توجه به فرض نخست ما که p > q است، با افزایش تعداد بلوک‌هایی که مهاجم باید بگذراند تا به زنجیره‌ی درست برسد، احتمال به صورت نمایی کاهش پیدا می‌کند. با توجه به کم بودن احتمال موفقیت او، اگر نتواند از آغاز خیزشی شانسی به جلو داشته باشد، همان‌گونه که از زنجیره پشت‌تر می‌افتد، بختش ناچیز کم خواهد شد.

اکنون بررسی می‌کنیم که دریافت‌کننده‌ی یک تراکنش تازه چه اندازه باید صبر کند تا بتواند به اندازه‌ی کافی از ناممکن بودن ویرایش تراکنش توسط فرستنده اطمینان حاصل کند. فرض می‌کنیم که فرستنده یک مهاجم است که قصد دارد دریافت‌کننده را مدتی قانع کند

که پرداخت صورت گرفته است و سپس بعد از گذشت مدتی پرداخت را برای خود بازگشت بزند. در این هنگام دریافت‌کننده آگاه خواهد شد ولی فرستنده امیدوار است که کار از کار گذشته باشد.

دریافت‌کننده یک جفت کلید تازه می‌سازد و کلید عمومی را، کمی پیش از امضا، در اختیار فرستنده قرار می‌دهد. این کار از اینکه فرستنده از پیش، آنقدر کار کند تا با شانس اوردن به اندازه‌ی کافی جلو بیفتد و تراکنش را در آن لحظه انجام دهد و بدین گونه زنجیره‌ای از بلوک های جایگزین از پیش آماده کند، پیش‌گیری می‌کند. هنگامی که تراکنش فرستاده شد، فرستنده‌ی متقلب مخفیانه شروع به کار روی زنجیره‌ی موازی که حاوی نسخه‌ی جایگزین تراکنشش است، می‌کند.

گیرنده تا هنگامی که تراکنش به یک بلوک اضافه شود و تعداد z بلوک پس از آن زنجیر شوند، منتظر می‌ماند. او از میزان پیشرفت دقیق مهاجم ناآگاه است؛ ولی با این فرض که بلوک‌های درست زمان میانگینِ موردِ انتظار به ازای بلوک را طی کرده‌اند، پیشرفت بالقوه مهاجم یک پراکندگی پواسون است با این مقدار مورد انتظار:

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

اکنون برای برآورد احتمال اینکه مهاجم هنوز بتواند به زنجیره‌ی درست برسد، چگالی پواسون، به ازای هر مقدار پیشرفتی که مهاجم ممکن است داشته باشد، را در احتمالِ رسیدن از آن نقطه به زنجیره‌ی درست، ضرب می‌کنیم:

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

برای پیشگیری از جمع کردن انتهای بی‌نهایتِ پراکندگی، جابجا می‌کنیم:

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

به کد C تبدیل می‌کنیم:

```
#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;
}
```

نتیجه‌هایی را برآورد می‌کنیم؛ می‌توان دید احتمال با افزایش z، نمایی کاهش می‌یابد:

```
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
```

با فرض P کمتر از 0.1%، بدست می‌آوریم...

```
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
```

۱۲- نتیجه

ما سامانه‌ای برای تراکنش‌های الکترونیکی پیشنهاد کرده‌ایم که به اعتماد متکی نیست. با چهارچوب آشنای سکه‌های ساخته شده از امضاهای دیجیتال، که کنترلِ مالکیتِ قوی‌ای فراهم می‌کند ولی بدون روشی برای پیش‌گیری از بازخرج پول ناکامل است، آغاز کردیم. برای حل این مشکل شبکه‌ای همتابه‌همتا را پیشنهاد کردیم که با استفاده از اثباتِ انجامِ کار تاریخچه‌ای همگانی از تراکنش‌ها ثبت می‌کند؛ که اگر گره‌های درست‌کار بیشترِ توانِ پردازشی را در دست داشته باشند، ویرایش در آن به‌سرعت برای یک مهاجم از نظر محاسباتی غیرعملی خواهد شد. شبکه با سادگی بی‌ساختار خود پایدار است. گره‌ها همگی همزمان و با کمترین هماهنگی کار می‌کنند. نیاز به شناسایی ندارند چرا که پیغام‌ها به مکان مشخصی هدایت نمی‌شوند و تنها نیاز است تا بر پایه بهترین تلاش رسانده شوند. گره‌ها، با پذیرش بلندترین زنجیره‌یِ اثباتِ کار به عنوان چیزی که در نبود آن‌ها رخ داده، می‌توانند آزادانه از شبکه بیرون بروند یا به آن بپیوندند. با توان پردازشی رای می‌دهند؛ پذیرش بلوک‌های معتبر را با کار بر روی بلندتر کردن آن‌ها و رد بلوک‌های نامعتبر را با کار نکردن بر روی آن‌ها، اعلام می‌کنند. هر گونه قانون و انگیزه‌ی لازم می‌تواند با این سازوکار همداستانی[۶] پیاده‌سازی شود.

یادکردها

[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.

۶ consensus