اتریوم: بستری نسل جدید برای قراردادهای هوشمند و برنامه‌های غیرمتمرکز

نویسنده: ویتالیک بوترین – (Vitalik Buterin) ۲۰۱۴

مترجم: مجتبی شفیعی (github.com/this-mojtaba)

زمانی که ساتوشی ناکاموتو (Satoshi Nakamoto) در ژانویهٔ ۲۰۰۹ بلاکچین بیت‌کوین را برای نخستین‌بار معرفی کرد، به‌طور هم‌زمان دو مفهوم رادیکال و آزمون‌نشده را ارائه کرد. مفهوم نخست بیت‌کوین (Bitcoin) بود؛ یک ارز آنلاین همتابه‌همتا (peer to peer) و غیرمتمرکز که بدون پشتوانه، بدون ارزش ذاتی و بدون صادرکنندهٔ مرکزی، ارزش خود را حفظ می‌کند. تاکنون بیت‌کوین به‌عنوان یک واحد پولی، بخش عمدهٔ توجه عمومی را به خود اختصاص داده است؛ هم از منظر سیاسیِ ارزی بدون بانک مرکزی و هم به‌دلیل نوسانات شدید قیمت آن.

با این حال، بخش دیگری از آزمایش بزرگ ساتوشی وجود دارد که به همان اندازه اهمیت دارد: مفهوم بلاکچینی مبتنی بر اثبات کار (proof of work) که امکان دستیابی به توافق عمومی بر أساس ترتیب تراکنش‌ها را فراهم می‌کند. بیت‌کوین را می‌توان یک سیستم (First-to-File) توصیف کرد؛ به این معنا که اگر یک موجودیت (کیف پول) ۵۰ بیت‌کوین داشته باشد و به‌طور هم‌زمان همان مقدار را به Aو B ارسال کند، تنها تراکنشی که زودتر تأیید شود پردازش خواهد شد. هیچ روش ذاتی‌ای برای تشخیص تقدم زمانی تراکنش‌ها وجود ندارد و همین مسئله برای دهه‌ها مانع توسعهٔ ارزهای دیجیتال غیرمتمرکز بود. بلاکچین ساتوشی نخستین راه‌حل غیرمتمرکز معتبر برای این مشکل به‌شمار می‌رفت. اکنون توجه به‌سرعت در حال معطوف‌شدن به این بخش از فناوری بیت‌کوین است و اینکه چگونه می‌توان از بلاکچین برای کاربردهایی فراتر از صرفاً پول استفاده کرد.

کاربردهایی که معمولاً به آن‌ها اشاره می‌شود شامل استفاده از دارایی‌های دیجیتال روی بلاکچین برای نمایش ارزهای سفارشی و ابزارهای مالی («کوین‌های رنگی»)، مالکیت دارایی‌های فیزیکی («مالکیت هوشمند»)، دارایی‌های غیرقابل‌تعویض مانند نام‌های دامنه (Namecoin) و همچنین کاربردهای پیشرفته‌تری مانند صرافی‌های غیرمتمرکز، مشتقات مالی، قمار (سیستم های بت مبتنی بر رمز ارز ها) همتابه‌همتا و سیستم‌های هویت و اعتبارسنجی هستند.

حوزهٔ مهم دیگری از پژوهش، قراردادهای هوشمند (Smart Contracts) است؛ سیستم‌هایی که دارایی‌های دیجیتال را بر اساس قواعد از پیش‌تعریف‌شده، به‌صورت خودکار جابه‌جا می‌کنند. برای مثال، می‌توان یک قرارداد خزانه‌داری (vault owner) تعریف کرد که در آن A بتواند روزانه تا سقف مشخصی برداشت کند، B نیز سقف برداشت جداگانه‌ای داشته باشد، Aو B با هم محدودیتی نداشته باشند و A بتواند دسترسی B را مسدود کند. گسترش منطقی این ایده، سازمان‌های خودگردان غیرمتمرکز (Decentralized Autonomous Organizations یا به اختصار DAO) هستند؛ قراردادهای هوشمند بلندمدتی که دارایی‌ها را در خود نگه می‌دارند و قوانین یک سازمان کامل را به‌صورت کد پیاده‌سازی می‌کنند.

آنچه اتریوم قصد ارائهٔ آن را دارد، بلاکچینی با یک زبان برنامه‌نویسی درونی و کامل، دارای توان محاسباتی تورینگ‌کامل-Turing) Complete) است که امکان تعریف توابع انتقال وضعیت دلخواه را فراهم می‌کند. این ویژگی به کاربران اجازه می‌دهد تنها با نوشتن منطق موردنظر خود در چند خط کد، تمام سیستم‌های توصیف‌شده ـ و بسیاری دیگر که هنوز حتی تصور نشده‌اند ـ را ایجاد کنند.

فهرست مطالب

مفهوم ارز دیجیتال غیرمتمرکز، و همچنین کاربردهای جایگزینی مانند ثبت مالکیت دارایی‌ها، دهه‌هاست که مطرح بوده است. پروتکل‌های ناشناس پول الکترونیکی (e-cash) در دهه‌های ۱۹۸۰ و ۱۹۹۰، که عمدتاً متکی بر یک سازوکار رمزنگاری به نام کورسازی چومی (Chaumian Blinding)بودند، ارزی با سطح بالایی از حریم خصوصی فراهم می‌کردند؛ اما این پروتکل‌ها به‌دلیل وابستگی به یک واسطهٔ متمرکز، در عمل نتوانستند به مورد پذیرش عموم دست یابند.

در سال ۱۹۹۸، طرح b-money که توسط وی دای (Wei Dai) ارائه شد، نخستین پیشنهاد برای معرفی ایدهٔ ایجاد پول از طریق حل معماهای محاسباتی و دستیابی به اجماع غیرمتمرکز بود؛ با این حال، این پیشنهاد در توضیح چگونگی پیاده‌سازی عملی اجماع غیرمتمرکز جزئیات کافی نداشت. در سال ۲۰۰۵، هال فینی (Hal Finney) مفهوم اثبات کار قابل‌استفادهٔ مجدد (Reusable Proofs of Work) را معرفی کرد؛ سیستمی که با ترکیب ایده‌های b-money و معماهای محاسباتی دشوار Hashcash ، که توسط آدام بک (Adam Back) ارائه شده بودند، چارچوبی برای یک ارز دیجیتال ایجاد می‌کرد، اما این طرح نیز به‌دلیل اتکا به محاسبات مورد اعتماد به‌عنوان لایهٔ پشتیبان، از دستیابی به یک راه‌حل ایده‌آل بازماند.

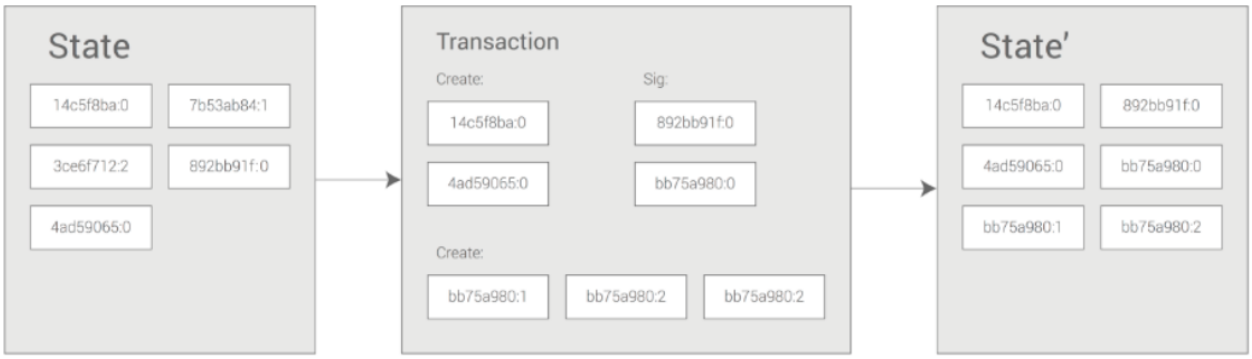
از آنجا که پول ذاتاً یک کاربرد «اولین ثبت‌شده» (first-file) است و ترتیب تراکنش‌ها در آن اهمیت حیاتی دارد، ارزهای دیجیتال غیرمتمرکز نیازمند راه‌حلی برای اجماع غیرمتمرکز هستند. مانع اصلی تمام پروتکل‌های پولی پیش از بیت‌کوین این بود که، با وجود سال‌ها پژوهش در زمینهٔ سیستم‌های اجماع چندطرفهٔ مقاوم در برابر خطای بیزانسی (Byzantine-Fault-Tolerant Systems)، این پروتکل‌ها تنها نیمی از مسئله را حل می‌کردند. آن‌ها فرض را بر این می‌گذاشتند که همهٔ شرکت‌کنندگان در سیستم شناخته‌شده هستند و حاشیه‌های امنیتی‌ای از این جنس تولید می‌کردند که اگر N طرف در سیستم حضور داشته باشند، سیستم می‌تواند تا N/4 بازیگر مخرب را تحمل کند.

اما در یک محیط ناشناس، چنین حاشیه‌های امنیتی‌ای در برابر حملات سیبیل (Sybil Attacks) آسیب‌پذیر هستند؛ حملاتی که در آن یک مهاجم با ایجاد هزاران گرهٔ شبیه‌سازی‌شده روی یک سرور یا بات‌نت، می‌تواند به‌صورت یک‌جانبه کنترل اکثریت را به دست بگیرد.

نوآوری‌ای که ساتوشی ارائه کرد، ترکیب یک پروتکل اجماع غیرمتمرکز بسیار ساده با سازوکار اثبات کار (Proof of Work) بود؛ به‌گونه‌ای که گره‌ها هر ده دقیقه تراکنش‌ها را در قالب یک بلاک گردآوری می‌کنند و یک زنجیرهٔ بلاک همواره در حال رشد ایجاد می‌شود. اثبات کار به‌عنوان مکانیزمی عمل می‌کند که از طریق آن، گره‌ها حق مشارکت در سیستم را به دست می‌آورند. هرچند گره‌هایی با توان محاسباتی بالا تأثیر بیشتری دارند، اما دستیابی به توانی بیش از کل شبکه، به‌مراتب دشوارتر از شبیه‌سازی یک میلیون گرهٔ جعلی است.

با وجود سادگی و خامی مدل بلاکچین بیت‌کوین، این مدل در عمل به‌اندازهٔ کافی خوب بوده است و طی پنج سال بعد، به زیربنای بیش از دویست ارز و پروتکل مختلف در سراسر جهان تبدیل شد.

بیت‌کوین به‌عنوان یک سیستم انتقال وضعیت



از دیدگاه فنی، دفترکل بیت‌کوین را می‌توان به‌عنوان یک سیستم انتقال وضعیت در نظر گرفت؛ سیستمی که در آن یک وضعیت وجود دارد که بیانگر مالکیت تمام بیت‌کوین‌های موجود است، و یک تابع انتقال وضعیت که با دریافت یک وضعیت و یک تراکنش، وضعیت جدیدی را به‌عنوان خروجی تولید می‌کند.

برای مثال، در یک سیستم بانکی متعارف، وضعیت برابر با ترازنامه است، تراکنش درخواستی برای انتقال مقدار X دلار از A به B محسوب می‌شود، و تابع انتقال وضعیت مقدار X را از حساب A کم کرده و به حساب B اضافه می‌کند. اگر موجودی حساب A از X کمتر باشد، تابع انتقال وضعیت یک خطا بازمی‌گرداند. بنابراین می‌توان این فرآیند را به‌صورت رسمی به شکل زیر تعریف کرد:

APPLY(S,TX) -> S' or ERROR

در سیستم بانکی تعریف‌شدهٔ فوق:

APPLY({Alice: $50, Bob: $50}, "send $20 from Alice to Bob") = {Alice: $30, Bob: $70}

اما در حالت زیر:

APPLY({Alice: $50, Bob: $50}, "send $70 from Alice to Bob") = ERROR

وضعیت در بیت‌کوین مجموعه‌ای از تمام سکه‌هایی است که ایجاد شده‌اند اما هنوز خرج نشده‌اند؛ که از نظر فنی خروجی‌های خرج‌نشدهٔ تراکنش (Unspent Transaction Outputs یا (UTXO نامیده می‌شوند. هر UTXO دارای یک مقدار مشخص و یک مالک است که مالکیت آن توسط یک آدرس ۲۰ بایتی تعریف می‌شود؛ آدرسی که در اصل یک کلید عمومی رمزنگاری‌شده است.

یک تراکنش شامل یک یا چند ورودی است که هر ورودی به یک UTXO موجود اشاره می‌کند و دارای یک امضای رمزنگاری‌شده است که با استفاده از کلید خصوصی متناظر با آدرس مالک تولید شده است. همچنین تراکنش می‌تواند شامل یک یا چند خروجی باشد که هر خروجی یک UTXO جدید را ایجاد کرده و به وضعیت شبکه افزوده می‌شود.

تابع انتقال وضعیت APPLY(S, TX) → S' را می‌توان به‌طور تقریبی به‌صورت زیر تعریف کرد:

1 .برای هر ورودی در تراکنش: TX

- اگر UTXO ارجاع‌داده‌شده در وضعیت S وجود نداشته باشد، یک خطا بازگردانده می‌شود.
- اگر امضای ارائه‌شده با مالک UTXO مطابقت نداشته باشد، یک خطا بازگردانده می‌شود.

2 .اگر مجموع مقادیر تمامUTXO های ورودی کمتر از مجموع مقادیر تمامUTXO های خروجی باشد، یک خطا بازگردانده می‌شود.

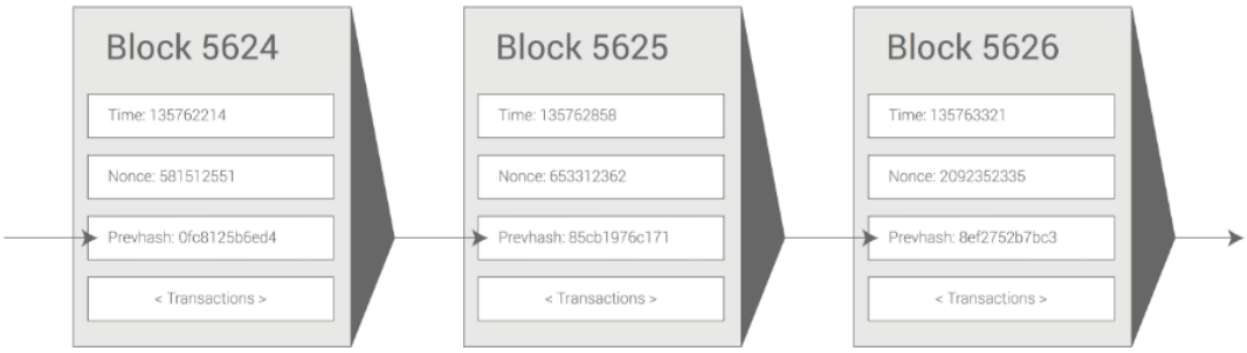
3 .در غیر این صورت، وضعیت S با حذف تمامUTXO های ورودی و افزودن تمامUTXO های خروجی بازگردانده می‌شود.

نیمهٔ نخست گام اول مانع از آن می‌شود که ارسال‌کنندهٔ تراکنش، سکه‌هایی را خرج کند که وجود ندارند؛ نیمهٔ دوم آن مانع خرج‌کردن سکه‌های دیگران می‌شود؛ و گام دوم، اصل بقای ارزش را تضمین می‌کند.

برای استفاده از این سازوکار در پرداخت، پروتکل به این صورت عمل می‌کند: فرض کنید آلیس قصد دارد ۱۱٫۷ بیت‌کوین به باب ارسال کند. ابتدا آلیس مجموعه‌ای ازUTXO های در دسترس خود را که مجموع آن‌ها دست‌کم ۱۱٫۷ بیت‌کوین باشد، انتخاب می‌کند. در عمل، معمولاً امکان دستیابی دقیق به همین مقدار وجود ندارد؛ برای مثال، فرض کنید کوچک‌ترین مجموعه‌ای که آلیس می‌تواند پیدا کند برابر با ۶ + ۴ + ۲ = ۱۲ بیت‌کوین باشد.

در این حالت، آلیس تراکنشی با سه ورودی و دو خروجی ایجاد می‌کند. خروجی نخست شامل ۱۱٫۷ بیت‌کوین با آدرس باب به‌عنوان مالک است، و خروجی دوم شامل ۰٫۳ بیت‌کوین باقی‌مانده به‌عنوان «باقی‌مانده» (Change) است که مالک آن خود آلیس خواهد بود.

استخراج



اگر به یک سرویس متمرکز و قابل اعتماد دسترسی داشتیم، پیاده‌سازی این سیستم بسیار ساده بود؛ چرا که می‌شد آن را دقیقاً مطابق توضیحات ارائه‌شده کدنویسی کرد. اما در بیت‌کوین، هدف ساخت یک سیستم پولی غیرمتمرکز است؛ بنابراین لازم است سیستم انتقال وضعیت با یک سیستم اجماع ترکیب شود تا اطمینان حاصل شود که همهٔ شرکت‌کنندگان بر سر ترتیب تراکنش‌ها به توافق می‌رسند.

اگر به یک سرویس متمرکز و قابل اعتماد دسترسی داشتیم، پیاده‌سازی این سیستم بسیار ساده بود؛ چرا که می‌شد آن را دقیقاً مطابق توضیحات ارائه‌شده کدنویسی کرد. اما در بیت‌کوین، هدف ساخت یک سیستم پولی غیرمتمرکز است؛ بنابراین لازم است سیستم انتقال وضعیت با یک سیستم اجماع ترکیب شود تا اطمینان حاصل شود که همهٔ شرکت‌کنندگان بر سر ترتیب تراکنش‌ها به توافق می‌رسند.

فرآیند اجماع غیرمتمرکز بیت‌کوین مستلزم آن است که گره‌های شبکه به‌طور مداوم تلاش کنند بسته‌هایی از تراکنش‌ها را تولید کنند که «بلاک» نامیده می‌شوند. شبکه به‌گونه‌ای طراحی شده است که به‌طور متوسط هر ده دقیقه یک بلاک تولید شود. هر بلاک شامل یک برچسب زمانی، یک نانس، یک ارجاع به بلاک قبلی — که در عمل همان هش بلاک پیشین است — و فهرستی از تمام تراکنش‌هایی است که از زمان بلاک قبلی انجام شده‌اند.

با گذشت زمان، این ساختار یک بلاکچین پایدار و همواره در حال رشد ایجاد می‌کند که به‌طور مداوم برای نمایش آخرین وضعیت دفترکل بیت‌کوین به‌روزرسانی می‌شود.

الگوریتم بررسی معتبر بودن یک بلاک، در این چارچوب، به‌صورت زیر تعریف می‌شود:

1 .بررسی شود که بلاک قبلی‌ای که بلاک فعلی به آن ارجاع می‌دهد وجود داشته و معتبر باشد.

2 .بررسی شود که برچسب زمانی بلاک از برچسب زمانی بلاک قبلی بزرگ‌تر بوده و بیش از دو ساعت جلوتر از زمان فعلی نباشد.

3 .بررسی شود که اثبات کار بلاک معتبر باشد.

4 .فرض می‌کنیم S[0] وضعیت شبکه در انتهای بلاک قبلی باشد.

5 .فرض کنید TX فهرست تراکنش‌های بلاک با n تراکنش باشد. برای تمام i در بازهٔ 0 تا n-1 ، مقدار S[i+1] را برابر با APPLY(S[i], TX[i]) قرار می‌دهیم. اگر اجرای هرکدام از این مراحل با خطا مواجه شود، بلاک نامعتبر تلقی می‌شود.

6 .در صورت موفقیت تمام مراحل، وضعیت نهایی S[n] به‌عنوان وضعیت شبکه در انتهای این بلاک ثبت می‌شود.

در اصل، هر تراکنش موجود در بلاک باید یک انتقال وضعیت معتبر ارائه دهد. وضعیت شبکه در خود بلاک ذخیره نمی‌شود و تنها با شروع از وضعیت اولیه و اعمال متوالی تمام تراکنش‌ها قابل محاسبه است.

همچنین ترتیب تراکنش‌ها در بلاک اهمیت دارد؛ اگر تراکنشی بخواهد خروجی ایجادشده توسط تراکنش دیگری را خرج کند، باید پس از آن در بلاک قرار گیرد.

بخش مهم الگوریتم اعتبارسنجی بلاک، اثبات کار است. شرط اصلی این است که هش SHA256 بلاک، به‌عنوان یک عدد ۲۵۶ بیتی، کمتر از یک مقدار هدف پویا باشد. این سازوکار باعث می‌شود تولید بلاک از نظر محاسباتی دشوار شود و بازنویسی زنجیره توسط مهاجمان ممکن نباشد.

از آنجا که SHA256 یک تابع شبه‌تصادفی غیرقابل‌پیش‌بینی است، تولید بلاک معتبر تنها از طریق آزمون و خطا امکان‌پذیر است. مقدار هدف به‌صورت دوره‌ای هر ۲۰۱۶ بلاک تنظیم می‌شود تا به‌طور متوسط هر ده دقیقه یک بلاک جدید تولید شود.

برای جبران این تلاش محاسباتی، استخراج‌کنندهٔ هر بلاک پاداشی برابر با ۲۵ بیت‌کوین دریافت می‌کند. همچنین کارمزد تراکنش‌ها نیز به استخراج‌کننده تعلق می‌گیرد. این تنها روش صدور بیت‌کوین‌های جدید است؛ چرا که وضعیت اولیه شبکه هیچ سکه‌ای نداشت.

برای درک بهتر هدف استخراج، بررسی می‌کنیم در صورت حضور یک مهاجم مخرب چه اتفاقی رخ می‌دهد. از آنجا که رمزنگاری زیربنایی بیت‌کوین امن در نظر گرفته می‌شود، مهاجم بخشی از سیستم بیت‌کوین را هدف قرار می‌دهد که به‌طور مستقیم توسط رمزنگاری محافظت نمی‌شود: ترتیب تراکنش‌ها.

راهبرد مهاجم ساده است:

1 .ارسال ۱۰۰ بیت‌کوین به یک فروشنده در ازای دریافت یک کالا (ترجیحاً یک کالای دیجیتال با تحویل سریع(

2 .انتظار برای دریافت کالا

3 .ایجاد یک تراکنش دیگر که همان ۱۰۰ بیت‌کوین را به خود مهاجم ارسال می‌کند

4 .تلاش برای قانع‌کردن شبکه به این‌که تراکنش دوم زودتر انجام شده است

پس از انجام گام نخست، یکی از استخراج‌کنندگان تراکنش را در بلاک شمارهٔ ۲۷۰۰۰۰ قرار می‌دهد. حدود یک ساعت بعد، پنج بلاک دیگر به زنجیره افزوده می‌شوند که به‌صورت غیرمستقیم تراکنش را تأیید می‌کنند. در این مرحله، فروشنده پرداخت را نهایی تلقی کرده و کالا را تحویل می‌دهد.

اکنون مهاجم تراکنش دیگری ایجاد می‌کند که همان ۱۰۰ بیت‌کوین را به خودش ارسال می‌کند. این تراکنش به‌طور عادی پردازش نمی‌شود، زیرا تلاش می‌کند یک UTXO خرج‌شده را دوباره مصرف کند.

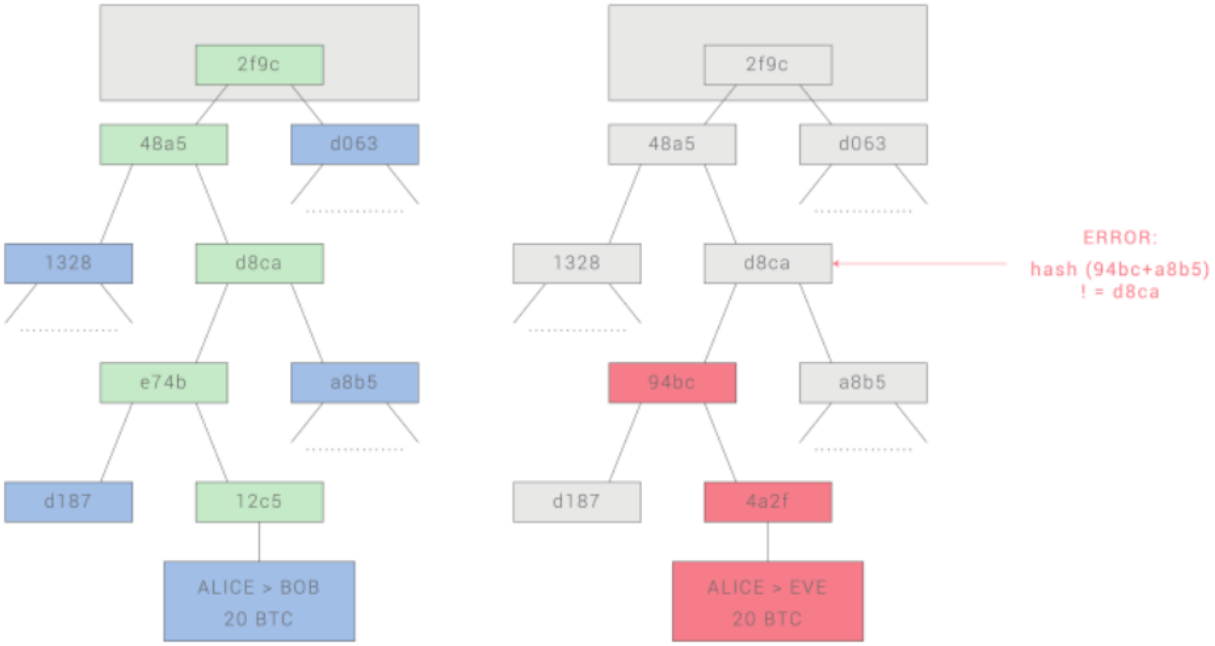
در نتیجه، مهاجم یک انشعاب از بلاکچین ایجاد می‌کند و نسخهٔ متفاوتی از بلاک ۲۷۰۰۰۰ را استخراج می‌کند. به دلیل تفاوت داده‌ها، انجام دوبارهٔ اثبات کار لازم است و زنجیرهٔ جدید از زنجیرهٔ اصلی جدا می‌شود.

طبق قاعدهٔ اجماع، طولانی‌ترین زنجیره به‌عنوان زنجیرهٔ معتبر پذیرفته می‌شود. استخراج‌کنندگان معتبر روی زنجیرهٔ اصلی کار می‌کنند و مهاجم به‌تنهایی روی زنجیرهٔ خود باقی می‌ماند. برای موفقیت مهاجم، لازم است توان محاسباتی‌ای بیش از کل شبکه داشته باشد؛ وضعیتی که به حملهٔ ۵۱٪ معروف است.

درخت‌های مرکل (Merkle Trees)

(سمت چپ) برای اثبات اعتبار یک شاخه در درخت مرکل، ارائهٔ تنها تعداد کمی از گره‌ها کافی است.

(سمت راست) هرگونه تلاش برای تغییر هر بخش از درخت مرکل، در نهایت به بروز یک ناسازگاری در جایی از زنجیره منجر می‌شود.



یکی از ویژگی‌های مهم بیت‌کوین از نظر مقیاس‌پذیری این است که بلاک‌ها در یک ساختار داده‌ای چندسطحی ذخیره می‌شوند. هش یک بلاک در واقع تنها هش سرآیند بلاک است؛ داده‌ای در حدود ۲۰۰ بایت که شامل برچسب زمانی، نانس، هش بلاک قبلی و هش ریشهٔ ساختاری به نام درخت مرکل است که تمام تراکنش‌های بلاک را در خود جای می‌دهد.

درخت مرکل نوعی درخت دودویی است که از گره‌های برگ در پایین، گره‌های میانی حاصل از هش فرزندان، و یک گرهٔ ریشه در بالاترین سطح تشکیل شده است.

هدف این ساختار آن است که داده‌های بلاک بتوانند به‌صورت بخش‌به‌بخش منتقل شوند. یک گره می‌تواند تنها سرآیند بلاک و بخش مرتبطی از درخت مرکل را دریافت کند و همچنان از صحت کل داده‌ها مطمئن باشد.

علت این موضوع آن است که تغییر در هر بخش از درخت، به‌طور زنجیره‌ای تا ریشه گسترش می‌یابد و در نهایت باعث تغییر هش بلاک می‌شود. به همین دلیل، هرگونه دست‌کاری منجر به شناسایی بلاک به‌عنوان یک بلاک نامعتبر خواهد شد.

درخت مرکل یکی از اجزای حیاتی برای پایداری بلندمدت شبکه است. گره‌های کامل که تمام بلاکچین را ذخیره می‌کنند، فضای ذخیره‌سازی زیادی نیاز دارند و این مقدار به‌مرور زمان افزایش می‌یابد.

پروتکل اعتبارسنجی سادهٔ پرداخت یا SPV، امکان استفاده از گره‌های سبک را فراهم می‌کند. این گره‌ها تنها سرآیند بلاک‌ها و شاخه‌های مرتبط با تراکنش‌های خود را دانلود می‌کنند و در عین حال می‌توانند با امنیت بالا وضعیت تراکنش‌ها و موجودی خود را بررسی کنند.

کاربردهای جایگزین بلاکچین

ایدهٔ استفاده از مفهوم زیربنایی بلاکچین و به‌کارگیری آن برای مفاهیمی فراتر از پول، سابقه‌ای طولانی دارد. در سال ۲۰۰۵، نیک سابو (Nick Szabo)مفهومی با عنوان «اسناد مالکیت امن با اختیار مالک (Secure Property Titles with Owner Authority) «را مطرح کرد؛ سندی که توضیح می‌داد چگونه «پیشرفت‌های جدید در فناوری پایگاه‌داده‌های تکرارشونده» می‌توانند امکان ایجاد سیستمی مبتنی بر بلاکچین را برای ثبت اینکه چه کسی مالک چه زمینی است فراهم کنند. این چارچوب شامل مفاهیمی مانند تصرف اولیه (Homesteading)، تصرف عدوانی (Adverse Possession) و مالیات زمین جورجی (Georgian Land Tax) بود. با این حال، متأسفانه در آن زمان هیچ سیستم پایگاه‌دادهٔ تکرارشوندهٔ مؤثری در دسترس نبود و به همین دلیل، این پروتکل هرگز در عمل پیاده‌سازی نشد.

اما پس از سال ۲۰۰۹، و با توسعهٔ اجماع غیرمتمرکز بیت‌کوین، تعدادی از کاربردهای جایگزین به‌سرعت شروع به ظهور کردند:

Namecoin

که در سال ۲۰۱۰ ایجاد شد، بهترین توصیف را می‌توان برای آن یک پایگاه‌دادهٔ غیرمتمرکز ثبت نام دانست. در پروتکل‌های غیرمتمرکزی مانندTor ، بیت‌کوین وBitMessage ، لازم است روشی برای شناسایی حساب‌ها وجود داشته باشد تا دیگران بتوانند با آن‌ها تعامل کنند؛ اما در تمام راه‌حل‌های موجود، تنها نوع شناسهٔ در دسترس یک هش شبه‌تصادفی است، مانند

LW79wp5ZBqaHW1jL5TCiBCrhQYtHagUWy 1

در حالت ایده‌آل، ترجیح داده می‌شود که حسابی با نامی ساده مانند «george» وجود داشته باشد. مشکل اینجاست که اگر یک نفر بتواند حسابی با نام «george» ایجاد کند، شخص دیگری نیز می‌تواند همان فرآیند را طی کرده و نام «george» را برای خود ثبت کند و خود را جای او جا بزند. تنها راه‌حل، استفاده از الگوی «اولین ثبت‌شده» است؛ به این معنا که اولین ثبت‌کننده موفق می‌شود و ثبت‌کنندهٔ دوم شکست می‌خورد. این دقیقاً مسئله‌ای است که پروتکل اجماع بیت‌کوین به‌خوبی برای حل آن مناسب است Namecoin .قدیمی‌ترین و موفق‌ترین پیاده‌سازی یک سیستم ثبت نام بر پایهٔ چنین ایده‌ای به شمار می‌رود.

کوین‌های رنگی

هدف کوین‌های رنگی فراهم‌کردن پروتکلی است که به افراد اجازه دهد ارزهای دیجیتال خود — یا در حالت ساده‌تر و مهم، توکن‌های دیجیتال — را روی بلاکچین بیت‌کوین ایجاد کنند. در پروتکل کوین‌های رنگی، یک ارز جدید از طریق اختصاص عمومی یک «رنگ» به یک UTXOمشخص بیت‌کوین صادر می‌شود. سپس پروتکل به‌صورت بازگشتی، رنگ سایرUTXO ها را برابر با رنگ ورودی‌هایی تعریف می‌کند که برای ایجاد آن‌ها خرج شده‌اند (در مواردی که ورودی‌ها رنگ‌های متفاوت دارند، قواعد خاصی اعمال می‌شود). این سازوکار به کاربران

اجازه می‌دهد کیف‌پول‌هایی داشته باشند که تنها شاملUTXO هایی با یک رنگ مشخص هستند و آن‌ها را مشابه بیت‌کوین‌های معمولی انتقال دهند؛ در حالی که برای تعیین رنگ هر UTXO دریافتی، بلاکچین به عقب پیمایش می‌شود.

متاکوین‌ها

ایدهٔ متاکوین بر ایجاد پروتکلی استوار است که بر بستر بیت‌کوین اجرا می‌شود؛ به این صورت که از تراکنش‌های بیت‌کوین برای ذخیرهٔ تراکنش‌های متاکوین استفاده می‌کند، اما تابع انتقال وضعیت متفاوتی با نام 'APPLY دارد. از آنجا که پروتکل متاکوین نمی‌تواند مانع از درج تراکنش‌های نامعتبر متاکوین در بلاکچین بیت‌کوین شود، قاعده‌ای اضافه می‌شود که اگر APPLY'(S, TX) با خطا مواجه شود، پروتکل به‌طور پیش‌فرض وضعیت را بدون تغییر نگه می‌دارد. این روش راهی ساده برای ایجاد یک پروتکل رمزارزی دلخواه فراهم می‌کند؛ حتی با ویژگی‌های پیشرفته‌ای که پیاده‌سازی آن‌ها در خود بیت‌کوین ممکن نیست، آن هم با هزینهٔ توسعهٔ بسیار پایین، چرا که پیچیدگی‌های استخراج و شبکه از پیش توسط پروتکل بیت‌کوین مدیریت شده‌اند.

به‌طور کلی، دو رویکرد اصلی برای ساخت یک پروتکل اجماع وجود دارد: ایجاد یک شبکهٔ مستقل، و ساخت یک پروتکل بر بستر بیت‌کوین. رویکرد نخست، اگرچه در مورد کاربردهایی مانند Namecoin تا حدی موفق بوده است، اما پیاده‌سازی دشواری دارد؛ زیرا هر پیاده‌سازی باید بلاکچین مستقل خود را راه‌اندازی کرده و تمام منطق انتقال وضعیت و کدهای شبکه را از ابتدا توسعه داده و آزمایش کند. افزون بر این، پیش‌بینی می‌شود که مجموعهٔ کاربردهای فناوری اجماع غیرمتمرکز از یک توزیع قانون توان پیروی کند؛ به این معنا که بخش عمده‌ای از کاربردها آن‌قدر کوچک خواهند بود که داشتن بلاکچین مستقل برای آن‌ها توجیه‌پذیر نباشد. همچنین دسته‌های بزرگی از کاربردهای غیرمتمرکز — به‌ویژه سازمان‌های خودگردان غیرمتمرکز — وجود دارند که نیازمند تعامل با یکدیگر هستند.

در مقابل، رویکرد مبتنی بر بیت‌کوین با این ضعف روبه‌رو است که ویژگی اعتبارسنجی سادهٔ پرداخت (SPV) بیت‌کوین را به ارث نمی‌برد . SPVدر بیت‌کوین کارآمد است، زیرا می‌تواند عمق بلاکچین را به‌عنوان معیاری برای اعتبار در نظر بگیرد؛ به این معنا که وقتی اجداد یک تراکنش به اندازهٔ کافی به عقب بازگردند، می‌توان با اطمینان گفت که آن تراکنش به‌طور معتبر بخشی از وضعیت شبکه بوده است. اما متاپروتکل‌های مبتنی بر بلاکچین نمی‌توانند بلاکچین را وادار کنند که از درج تراکنش‌هایی که در چارچوب پروتکل آن‌ها نامعتبر هستند جلوگیری کند. در نتیجه، یک پیاده‌سازی کاملاً امن از SPV برای متاپروتکل‌ها نیازمند پیمایش معکوس کل بلاکچین بیت‌کوین تا بلاک آغازین است تا بتوان اعتبار تراکنش‌ها را تعیین کرد.

در حال حاضر، تمام پیاده‌سازی‌های سبک متاپروتکل‌های مبتنی بر بیت‌کوین به یک سرور مورد اعتماد برای تأمین داده‌ها متکی هستند؛ نتیجه‌ای که می‌توان آن را به‌شدت نامطلوب دانست، به‌ویژه با توجه به این‌که یکی از اهداف اصلی رمزارزها حذف نیاز به اعتماد است.

اسکریپت‌نویسی

حتی بدون هیچ‌گونه افزونه‌ای، پروتکل بیت‌کوین در عمل نسخه‌ای ضعیف از مفهوم «قراردادهای هوشمند (Smart Contracts) «را فراهم می‌کند. در بیت‌کوین، خروجی‌های خرج‌نشدهٔ تراکنش‌ها یاUTXO ها می‌توانند نه‌تنها تحت مالکیت یک کلید عمومی، بلکه تحت

مالکیت یک اسکریپت پیچیده‌تر نیز قرار گیرند؛ اسکریپتی که در قالب یک زبان برنامه‌نویسی ساده و مبتنی بر پشته (Stack-Based) بیان می‌شود. در این چارچوب، تراکنشی که قصد خرج‌کردن یک UTXO را دارد، باید داده‌ای ارائه دهد که شرایط آن اسکریپت را برآورده کند.

در واقع، حتی سازوکار پایهٔ مالکیت مبتنی بر کلید عمومی نیز از طریق یک اسکریپت پیاده‌سازی شده است: اسکریپت، یک امضای مبتنی بر منحنی بیضوی را به‌عنوان ورودی دریافت می‌کند، آن را با تراکنش و آدرسی که مالک UTXO است تطبیق می‌دهد و در صورت موفقیت‌آمیز بودن اعتبارسنجی، مقدار ۱ و در غیر این صورت مقدار ۰ بازمی‌گرداند. افزون بر این، اسکریپت‌های پیچیده‌تری نیز برای کاربردهای متنوع دیگر وجود دارند.

برای مثال، می‌توان اسکریپتی ساخت که برای معتبرشناخته‌شدن، نیازمند امضای دو مورد از سه کلید خصوصی مشخص باشد («چندامضایی» یا(Multisig ؛ ساختاری که برای حساب‌های سازمانی، حساب‌های پس‌انداز امن و برخی سناریوهای امانت‌داری تجاری کاربرد دارد. همچنین از اسکریپت‌ها می‌توان برای پرداخت پاداش در ازای حل مسائل محاسباتی استفاده کرد، و حتی می‌توان اسکریپتی تعریف کرد که چیزی شبیه به این بگوید: «این UTXO بیت‌کوین متعلق به توست، اگر بتوانی یک اثبات SPV ارائه دهی که نشان دهد تراکنشی از این مقدار در شبکهٔ دوج‌کوین برای من ارسال کرده‌ای»؛ قابلیتی که عملاً امکان تبادل غیرمتمرکز میان رمزارزهای مختلف را فراهم می‌کند.

با این حال، زبان اسکریپت‌نویسی بیت‌کوین در شکل فعلی خود دارای محدودیت‌های مهمی است:

نخست، نبودِ تورینگ‌کامل‌بودن .(Turing-Completeness) به این معنا که هرچند زبان اسکریپت بیت‌کوین از مجموعهٔ بزرگی از محاسبات پشتیبانی می‌کند، اما تقریباً همهٔ آن‌ها را پوشش نمی‌دهد. مهم‌ترین قابلیتِ غایب، حلقه‌ها هستند. این محدودیت به‌منظور جلوگیری از ایجاد حلقه‌های بی‌نهایت در زمان اعتبارسنجی تراکنش‌ها اعمال شده است. از نظر تئوری، این مانع قابل دورزدن است؛ چرا که هر حلقه‌ای را می‌توان با تکرار دستی کد و استفاده از شرط‌ها شبیه‌سازی کرد، اما در عمل این کار منجر به اسکریپت‌هایی بسیار ناکارآمد از نظر حجم می‌شود. برای نمونه، پیاده‌سازی یک الگوریتم جایگزین برای امضای مبتنی بر منحنی بیضوی احتمالاً نیازمند ۲۵۶ مرحلهٔ ضرب تکرارشونده است که هرکدام باید به‌صورت جداگانه در کد گنجانده شوند.

دوم، ناآگاهی از مقدار .(Value-Blindness) اسکریپت‌های UTXO هیچ راهی برای اعمال کنترل دقیق بر مقدار قابل برداشت ندارند. برای مثال، یکی از کاربردهای قدرتمند قراردادهای مبتنی بر اوراکل می‌تواند یک قرارداد پوشش ریسک باشد؛ جایی که A و B هرکدام معادل ۱۰۰۰ دلار بیت‌کوین واریز می‌کنند و پس از ۳۰ روز، اسکریپت ۱۰۰۰ دلار بیت‌کوین به A و باقی‌مانده را به B ارسال می‌کند. این سازوکار مستلزم وجود یک اوراکل برای تعیین ارزش بیت‌کوین نسبت به دلار است، اما حتی در این حالت نیز از نظر نیاز به اعتماد و زیرساخت، پیشرفتی چشمگیر نسبت به راه‌حل‌های کاملاً متمرکز فعلی به شمار می‌رود. با این حال، از آنجا کهUTXO ها ماهیتی «همه یا هیچ» دارند، تنها راه دستیابی به چنین رفتاری استفاده از روشی بسیار ناکارآمد است؛ یعنی ایجاد تعداد زیادی UTXO با مقادیر متفاوت (مثلاً یک UTXO به ارزش ۲۰۰۰ واحد برای هر k تا ۳۰) و واگذاری تصمیم به اوراکل برای انتخاب اینکه کدام UTXO به A و کدام به B ارسال شود.

سوم، نبودِ وضعیت .(State) یک UTXO یا خرج می‌شود یا خرج‌نشده باقی می‌ماند و هیچ امکان دیگری برای نگه‌داری وضعیت داخلی وجود ندارد. این موضوع ساخت قراردادهای چندمرحله‌ای، پیشنهادهای صرافی غیرمتمرکز یا پروتکل‌های تعهد رمزنگاری دومرحله‌ای — که برای پاداش‌های محاسباتی امن ضروری هستند — را دشوار می‌کند. همچنین این محدودیت به این معناست کهUTXO ها تنها برای قراردادهای ساده و یک‌بارمصرف مناسب‌اند و نمی‌توان از آن‌ها برای قراردادهای «دارای وضعیت» پیچیده‌تری مانند سازمان‌های غیرمتمرکز استفاده کرد؛ امری که پیاده‌سازی متاپروتکل‌ها را نیز با مشکل مواجه می‌سازد. ترکیب وضعیت دودویی با ناآگاهی از مقدار، باعث می‌شود یکی دیگر از کاربردهای مهم، یعنی تعیین سقف برداشت، عملاً غیرممکن باشد.

چهارم، ناآگاهی از بلاکچین .(Blockchain-Blindness) اسکریپت‌های UTXO به داده‌های بلاکچین مانند نانس یا هش بلاک قبلی دسترسی ندارند. این موضوع کاربردهایی مانند قمار و چندین دستهٔ دیگر را به‌شدت محدود می‌کند، چرا که زبان اسکریپت را از یک منبع بالقوهٔ ارزشمند برای تولید تصادفی‌بودن محروم می‌سازد.

در نتیجه، می‌توان سه رویکرد اصلی برای ساخت کاربردهای پیشرفته بر بستر رمزارزها شناسایی کرد: ساخت یک بلاکچین جدید، استفاده از اسکریپت‌نویسی بر بستر بیت‌کوین، و ایجاد یک متاپروتکل بر روی بیت‌کوین. ساخت بلاکچین جدید آزادی کامل در طراحی قابلیت‌ها را فراهم می‌کند، اما مستلزم صرف زمان توسعه و تلاش فراوان برای راه‌اندازی اولیه است. استفاده از اسکریپت‌نویسی ساده و استانداردپذیر است، اما به‌شدت از نظر قابلیت محدود است؛ و متاپروتکل‌ها، هرچند پیاده‌سازی ساده‌ای دارند، از مشکلات جدی در مقیاس‌پذیری رنج می‌برند. هدف اتریوم آن است که چارچوبی عمومی ایجاد کند که مزایای هر سه رویکرد را به‌طور هم‌زمان در اختیار بگذارد.

اتریوم

هدف اتریوم آن است که مفاهیم اسکریپت‌نویسی، آلت‌کوین‌ها و متاپروتکل‌های درون‌زنجیره‌ای را با یکدیگر ادغام کرده و بهبود دهد، و به توسعه‌دهندگان امکان دهد برنامه‌های دلخواه مبتنی بر اجماع ایجاد کنند؛ برنامه‌هایی که به‌طور هم‌زمان از مقیاس‌پذیری، استانداردسازی، کامل‌بودن قابلیت‌ها، سهولت توسعه و تعامل‌پذیری‌ای برخوردار باشند که این رویکردهای مختلف ارائه می‌دهند.

اتریوم این کار را با ساخت چیزی انجام می‌دهد که در اصل می‌توان آن را «نهایی‌ترین لایهٔ زیربنایی انتزاعی» دانست: یک بلاکچین با یک زبان برنامه‌نویسی تورینگ‌کاملِ درونی، که به هر کسی اجازه می‌دهد قراردادهای هوشمند و برنامه‌های غیرمتمرکزی بنویسد که در آن‌ها می‌توان قواعد دلخواهی برای مالکیت، قالب تراکنش‌ها و توابع انتقال وضعیت تعریف کرد.

یک نسخهٔ حداقلی از Namecoin را می‌توان تنها در دو خط کد نوشت، و سایر پروتکل‌ها مانند سیستم‌های ارزی و سیستم‌های اعتبارسنجی را می‌توان با کمتر از بیست خط کد ساخت. همچنین می‌توان قراردادهای هوشمند را — «جعبه‌های رمزنگاری‌شده‌ای» که ارزش را در خود نگه می‌دارند و فقط در صورت برآورده‌شدن شرایط مشخصی آن را آزاد می‌کنند — بر بستر پلتفرم ما ایجاد کرد؛ قراردادهایی با قدرتی بسیار بیشتر از آنچه اسکریپت‌نویسی بیت‌کوین ارائه می‌دهد، به‌دلیل قابلیت‌های افزوده‌ای مانند تورینگ‌کامل‌بودن، آگاهی از مقدار، آگاهی از بلاکچین و داشتن وضعیت.

حساب‌های اتریوم

در اتریوم، وضعیت شبکه از مجموعه‌ای از اشیاء به نام «حساب‌ها (Accounts)» تشکیل شده است. هر حساب دارای یک آدرس ۲۰ بایتی است و انتقال وضعیت‌ها به‌صورت جابه‌جایی مستقیم مقدار و اطلاعات میان حساب‌ها انجام می‌شود. هر حساب اتریوم شامل چهار بخش است:

- نانس :(Nonce) یک شمارنده که برای اطمینان از این‌که هر تراکنش فقط یک‌بار پردازش شود استفاده می‌شود.
- موجودی فعلی اتر حساب.
- کد قرارداد حساب، در صورتی که وجود داشته باشد.
- فضای ذخیره‌سازی حساب (که به‌صورت پیش‌فرض خالی است).

اتر (Ether) سوخت رمزنگاری‌شدهٔ اصلی و داخلی اتریوم است و برای پرداخت کارمزد تراکنش‌ها استفاده می‌شود. به‌طور کلی، دو نوع حساب در اتریوم وجود دارد: حساب‌های تحت مالکیت خارجی، که توسط کلیدهای خصوصی کنترل می‌شوند، و حساب‌های قراردادی، که توسط کد قرارداد خود کنترل می‌شوند.

یک حساب تحت مالکیت خارجی فاقد کد است و می‌توان با ایجاد و امضای یک تراکنش، از چنین حسابی پیام ارسال کرد. در مقابل، در یک حساب قراردادی، هر بار که حساب یک پیام دریافت می‌کند، کد آن فعال می‌شود؛ به‌گونه‌ای که می‌تواند فضای ذخیره‌سازی داخلی را بخواند و بنویسد، پیام‌های دیگری ارسال کند یا به نوبهٔ خود قراردادهای جدیدی ایجاد کند.

پیام‌ها و تراکنش‌ها

»پیام‌ها (Messages) «در اتریوم تا حدی شبیه به «تراکنش‌ها» در بیت‌کوین هستند، اما سه تفاوت مهم دارند. نخست این‌که یک پیام در اتریوم می‌تواند هم توسط یک موجودیت خارجی و هم توسط یک قرارداد ایجاد شود، در حالی که تراکنش‌های بیت‌کوین تنها از سوی موجودیت‌های خارجی قابل ایجاد هستند. دوم این‌که پیام‌های اتریوم به‌طور صریح می‌توانند شامل داده باشند. و در نهایت، گیرندهٔ یک پیام در اتریوم — در صورتی که یک حساب قراردادی باشد — این امکان را دارد که پاسخی بازگرداند؛ به این معنا که پیام‌های اتریوم مفهوم «تابع» را نیز در بر می‌گیرند.

اصطلاح «تراکنش» در اتریوم برای اشاره به یک بستهٔ دادهٔ امضاشده به‌کار می‌رود که حاوی پیامی است که از یک حساب تحت مالکیت خارجی ارسال می‌شود. تراکنش‌ها شامل گیرندهٔ پیام، امضایی برای شناسایی فرستنده، مقدار اتر ارسالی و دادهٔ همراه پیام هستند؛ همچنین دو مقدار دیگر به نام‌های STARTGAS و GASPRICE را نیز در خود دارند.

برای جلوگیری از رشد نمایی محاسبات و ایجاد حلقه‌های بی‌نهایت در کد، هر تراکنش ملزم است محدودیتی برای تعداد گام‌های محاسباتی قابل اجرا تعیین کند؛ این محدودیت شامل هم پیام اولیه و هم پیام‌های اضافی‌ای می‌شود که در طول اجرای کد ایجاد می‌شوند. مقدار STARTGASهمین سقف را مشخص می‌کند و GASPRICE کارمزدی است که به‌ازای هر گام محاسباتی به استخراج‌کننده پرداخت می‌شود.

اگر اجرای یک تراکنش «گاز تمام کند(run out of gas) «، تمام تغییرات وضعیت لغو می‌شوند — به‌جز پرداخت کارمزدها. اما اگر اجرای تراکنش با مقداری گاز باقی‌مانده متوقف شود، بخش استفاده‌نشدهٔ کارمزد به فرستنده بازگردانده می‌شود.

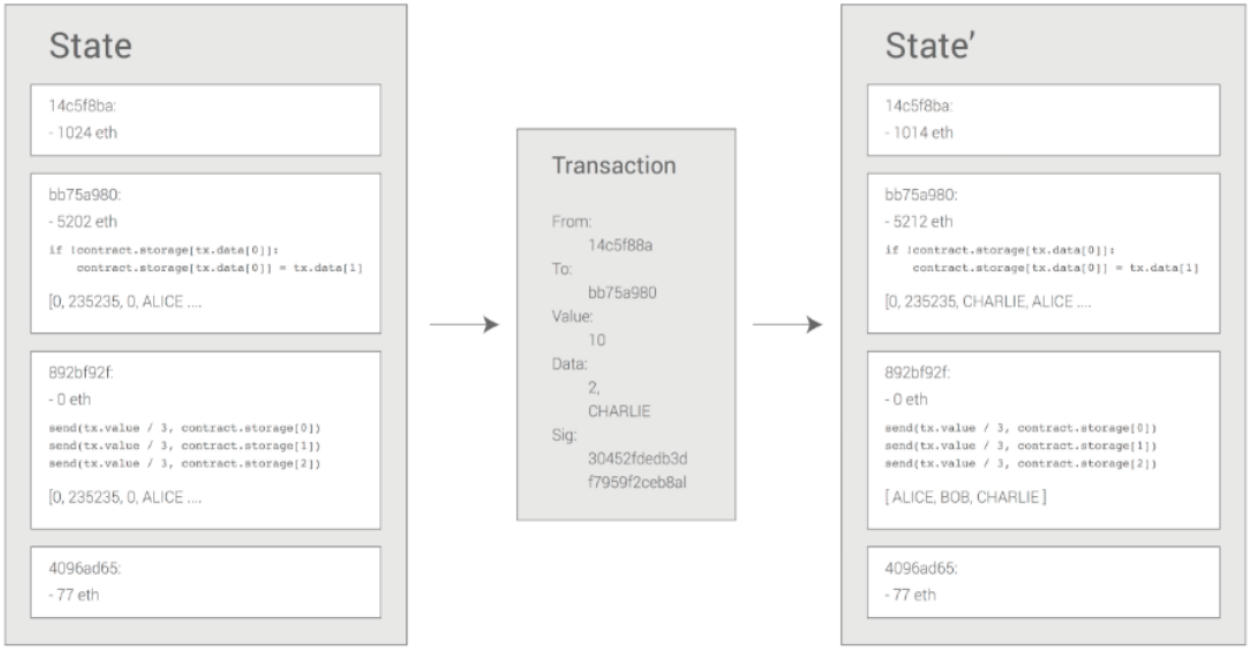
همچنین، یک نوع تراکنش جداگانه و پیام متناظر با آن برای ایجاد قرارداد وجود دارد؛ آدرس یک قرارداد بر اساس هش نانس حساب و داده‌های تراکنش محاسبه می‌شود.

یکی از پیامدهای مهم سازوکار پیام‌رسانی در اتریوم، ویژگی «شهروند درجه‌یک (First-Class Citizen) «بودن است؛ ایده‌ای که بر اساس آن، قراردادها از نظر توانمندی، هم‌سطح با حساب‌های خارجی هستند و از جمله می‌توانند پیام ارسال کنند یا قراردادهای دیگری ایجاد نمایند.

این ویژگی به قراردادها اجازه می‌دهد که به‌طور هم‌زمان نقش‌های بسیار متنوعی ایفا کنند. برای مثال، ممکن است یکی از اعضای یک سازمان غیرمتمرکز (یک قرارداد)، به‌عنوان یک حساب امانت‌دار (قراردادی دیگر) میان یک فرد بیش‌ازحد محتاط که از امضاهای لمپورت مقاوم در برابر محاسبات کوانتومی استفاده می‌کند (قراردادی سوم) و یک نهاد هم‌امضاکننده که خود از حسابی با پنج کلید برای امنیت بیشتر بهره می‌برد (قراردادی چهارم) عمل کند.

نقطهٔ قوت پلتفرم اتریوم در این است که سازمان غیرمتمرکز و قرارداد امانت‌دار نیازی ندارند بدانند هر یک از طرفین قرارداد از چه نوع حسابی استفاده می‌کند.

تابع انتقال وضعیت اتریوم



تابع انتقال وضعیت در اتریوم که با APPLY(S, TX) → S' نمایش داده می‌شود، به‌صورت زیر تعریف می‌گردد:

بررسی می‌شود که تراکنش به‌درستی شکل گرفته باشد (برای مثال تعداد مقادیر صحیح باشد)، امضا معتبر باشد و نانس تراکنش با نانس موجود در حساب فرستنده مطابقت داشته باشد. در غیر این صورت، خطا بازگردانده می‌شود.

کارمزد تراکنش به‌صورت STARTGAS × GASPRICE محاسبه می‌شود و آدرس فرستنده از روی امضا تعیین می‌گردد. این مقدار از موجودی حساب فرستنده کسر شده و نانس فرستنده یک واحد افزایش می‌یابد. اگر موجودی کافی نباشد، خطا بازگردانده می‌شود.

مقدار GAS برابر با STARTGAS مقداردهی اولیه می‌شود و به‌ازای هر بایت از داده‌های تراکنش، مقدار مشخصی گاز کسر می‌گردد تا هزینهٔ داده‌های موجود در تراکنش پرداخت شود.

مقدار اتر ارسالی تراکنش از حساب فرستنده به حساب گیرنده منتقل می‌شود. اگر حساب گیرنده هنوز وجود نداشته باشد، ایجاد می‌شود. اگر حساب گیرنده یک حساب قراردادی باشد، کد قرارداد اجرا می‌شود؛ یا تا پایان اجرا، یا تا زمانی که گاز تمام شود.

اگر انتقال مقدار به دلیل کافی نبودن موجودی فرستنده شکست بخورد، یا اجرای کد به دلیل اتمام گاز متوقف شود، تمام تغییرات وضعیت — به‌جز پرداخت کارمزد — بازگردانده می‌شوند و کارمزدها به حساب استخراج‌کننده افزوده می‌شوند.

در غیر این صورت، کارمزد مربوط به گاز استفاده‌نشده به فرستنده بازگردانده می‌شود و کارمزد گاز مصرف‌شده به استخراج‌کننده پرداخت می‌گردد.

برای مثال، فرض کنید کد قرارداد به‌شکل زیر باشد:

```
if !contract.storage[msg.data[0]]:
    contract.storage[msg.data[0]] = msg.data[1]
```

توجه داشته باشید که در عمل، کد قراردادها به زبان سطح پایین ماشین مجازی اتریوم (EVM) نوشته می‌شوند؛ این مثال برای شفافیت، به زبان سطح‌بالای Serpent نوشته شده است و قابل تبدیل به کد EVM است.

فرض کنید فضای ذخیره‌سازی قرارداد در ابتدا خالی است و یک تراکنش با مشخصات زیر ارسال می‌شود:

مقدار اتر: ۱۰ , گاز (gass) : ۲۰۰۰ , قیمت گاز: ۰٫۰۰۱ اتر ,داده‌ها: [2, ‘CHARLIE’]

فرآیند اجرای تابع انتقال وضعیت در این حالت به‌صورت زیر خواهد بود:

بررسی می‌شود که تراکنش معتبر و به‌درستی شکل گرفته باشد.

بررسی می‌شود که فرستنده حداقل 2 = 0.001 × 2000 اتر برای پرداخت کارمزد داشته باشد؛ در این صورت، ۲ اتر از حساب او کسر می‌شود.

مقدار گاز برابر با ۲۰۰۰ مقداردهی می‌شود. با فرض این‌که طول تراکنش ۱۷۰ بایت و هزینهٔ هر بایت ۵ باشد، مقدار ۸۵۰ گاز کسر می‌شود و ۱۱۵۰ گاز باقی می‌ماند.

مقدار ۱۰ اتر دیگر از حساب فرستنده کسر شده و به حساب قرارداد افزوده می‌شود.

کد قرارداد اجرا می‌شود. در این مثال، قرارداد بررسی می‌کند که آیا فضای ذخیره‌سازی در اندیس ۲ استفاده شده است یا خیر؛ چون خالی است، مقدار CHARLIE در آن ذخیره می‌شود. فرض می‌کنیم این عملیات ۱۸۷ گاز مصرف کند، در نتیجه مقدار گاز باقی‌مانده برابر با ۹۶۳ خواهد بود.

مقدار 0.963 = 0.001 × 963 اتر به حساب فرستنده بازگردانده می‌شود و وضعیت نهایی بازگشت داده می‌شود.

اگر در سمت گیرندهٔ تراکنش هیچ قراردادی وجود نداشت، کارمزد کل تراکنش صرفاً برابر با GASPRICE ضربدر طول تراکنش (بر حسب بایت) می‌بود و داده‌های همراه تراکنش اهمیتی نداشتند.

همچنین، پیام‌هایی که توسط قراردادها ایجاد می‌شوند می‌توانند برای محاسباتی که راه‌اندازی می‌کنند، محدودیت گاز تعیین کنند. اگر این محاسبات فرعی با اتمام گاز مواجه شوند، تنها تا نقطهٔ فراخوانی پیام بازگردانده می‌شوند. بنابراین، همانند تراکنش‌ها، قراردادها نیز می‌توانند با تعیین سقف‌های سخت‌گیرانه، منابع محاسباتی محدود خود را ایمن کنند.

اجرای کد

کد در قراردادهای اتریوم به یک زبان بایت‌کد سطح پایین و مبتنی بر پشته نوشته می‌شود که به آن «کد ماشین مجازی اتریوم « (Ethereum Virtual Machine Code)یا (EVM Code گفته می‌شود. این کد از یک رشته بایت تشکیل شده است که هر بایت نمایانگر یک عملیات است. به‌طور کلی، اجرای کد یک حلقهٔ بی‌نهایت است که در آن، عملیاتِ مربوط به شمارندهٔ فعلی برنامه (Program Counter)— که از صفر شروع می‌شود — به‌صورت پی‌درپی اجرا می‌گردد و سپس شمارندهٔ برنامه یک واحد افزایش می‌یابد؛ تا زمانی که به انتهای کد برسیم یا یک خطا رخ دهد یا دستور STOP یا RETURN مشاهده شود.

عملیات‌ها به سه نوع فضا برای ذخیرهٔ داده دسترسی دارند:

پشته :(Stack) یک ساختار آخرین‌ورود-اولین‌خروج (Last-In-First-Out) که مقادیر ۳۲ بایتی می‌توانند در آن قرار داده شوند یا از آن برداشته شوند.

حافظه :(Memory) یک آرایهٔ بایتی با قابلیت گسترش نامحدود.

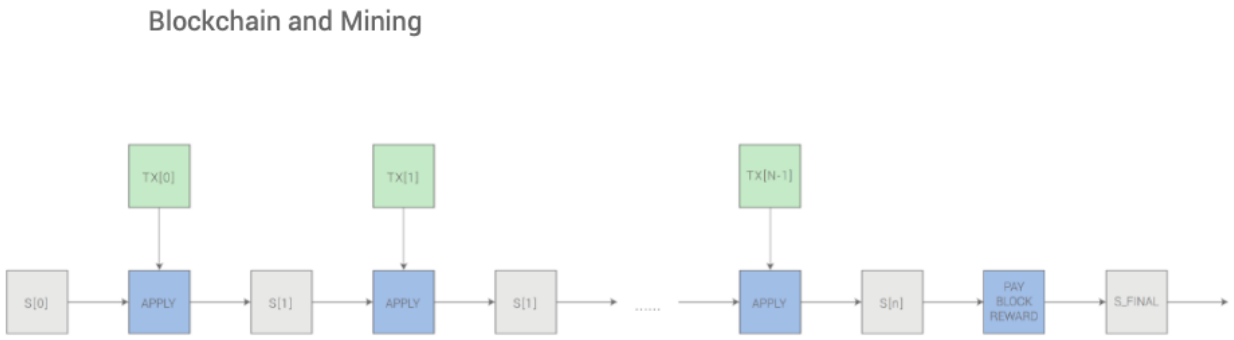
فضای ذخیره‌سازی بلندمدت قرارداد :(Storage) یک انبار کلید/مقدار که کلیدها و مقدارها هر دو ۳۲ بایتی هستند. برخلاف پشته و حافظه که پس از پایان محاسبه بازنشانی می‌شوند، ذخیره‌سازی در بلندمدت باقی می‌ماند.

کد همچنین می‌تواند به مقدار، فرستنده و دادهٔ پیام ورودی دسترسی داشته باشد، و نیز داده‌های سرآیند بلاک را بخواند؛ همچنین کد می‌تواند در خروجی، یک آرایهٔ بایتی از داده را بازگرداند.

مدل اجرایی رسمی کد EVM به شکل شگفت‌آوری ساده است. هنگامی که ماشین مجازی اتریوم در حال اجراست، کل وضعیت محاسباتی آن را می‌توان با یک تاپل به‌شکل (block_state, transaction, message, code, memory, stack, pc, gas) تعریف کرد؛ که در آن block_stateوضعیت جهانی شامل تمام حساب‌ها است و موجودی‌ها و ذخیره‌سازی را نیز در بر می‌گیرد. در هر دور از اجرا، دستور فعلی با گرفتن بایتِ شمارهٔ pc از کد مشخص می‌شود و هر دستور تعریف خاص خود را دارد که بیان می‌کند چگونه این تاپل را تغییر می‌دهد.

برای مثال، دستور ADD دو مقدار را از پشته برمی‌دارد و حاصل جمع آن‌ها را روی پشته قرار می‌دهد، گاز را به میزان ۱ کاهش می‌دهد و pc را یک واحد افزایش می‌دهد. دستور SSTORE نیز دو مقدار بالایی پشته را برمی‌دارد و مقدار دوم را در فضای ذخیره‌سازی قرارداد، در اندیسی که مقدار اول مشخص کرده است قرار می‌دهد؛ همچنین می‌تواند تا ۲۰۰ واحد گاز را کاهش دهد و pc را یک واحد افزایش دهد. با این‌که راه‌های زیادی برای بهینه‌سازی اتریوم با استفاده از کامپایل در لحظه (Just-In-Time Compilation) وجود دارد، یک پیاده‌سازی پایه از اتریوم را می‌توان با چند صد خط کد انجام داد.

بلاکچین و استخراج



بلاکچین اتریوم از بسیاری جهات به بلاکچین بیت‌کوین شباهت دارد، هرچند تفاوت‌هایی نیز میان آن‌ها وجود دارد. مهم‌ترین تفاوت میان اتریوم و بیت‌کوین از نظر معماری بلاکچین این است که، برخلاف بیت‌کوین، بلاک‌های اتریوم علاوه بر فهرست تراکنش‌ها، شامل نسخه‌ای از جدیدترین وضعیت شبکه نیز هستند. افزون بر این، دو مقدار دیگر یعنی شمارهٔ بلاک و درجهٔ سختی (Difficulty) نیز در هر بلاک ذخیره می‌شوند.

الگوریتم اعتبارسنجی بلاک در اتریوم به‌صورت زیر است:

1. بررسی می‌شود که بلاک قبلی‌ای که به آن ارجاع داده شده وجود داشته و معتبر باشد.
2. بررسی می‌شود که برچسب زمانی بلاک از برچسب زمانی بلاک قبلی بزرگ‌تر بوده و بیش از ۱۵ دقیقه جلوتر از زمان فعلی نباشد.
3. بررسی می‌شود که شمارهٔ بلاک، درجهٔ سختی، ریشهٔ تراکنش‌ها(Transaction Root) ، ریشهٔ بلاک‌های عمو (Uncle Root) و محدودیت گاز — (Gas Limit) که همگی مفاهیم سطح پایین مخصوص اتریوم هستند — معتبر باشند.
4. بررسی می‌شود که اثبات کار (Proof of Work) بلاک معتبر باشد.
5. مقدار S[0] برابر با ریشهٔ وضعیت (STATE_ROOT) بلاک قبلی در نظر گرفته می‌شود.
6. فرض می‌کنیم TX فهرست تراکنش‌های بلاک با n تراکنش باشد. برای تمام i در بازهٔ 0 تا n-1 ، مقدار S[i+1] برابر با APPLY(S[i],TX[i])قرار داده می‌شود. اگر اجرای هر مرحله با خطا مواجه شود، یا مجموع گاز مصرف‌شده در بلاک تا این نقطه از GASLIMITفراتر رود، بلاک نامعتبر تلقی می‌شود.
7. مقدار S_FINAL برابر با S[n] در نظر گرفته می‌شود، با این تفاوت که پاداش بلاک پرداخت‌شده به استخراج‌کننده نیز به آن افزوده می‌شود.
8. بررسی می‌شود که آیا S_FINAL با STATE_ROOT ذخیره‌شده در بلاک یکسان است یا خیر. اگر یکسان باشد، بلاک معتبر است؛ در غیر این صورت، نامعتبر خواهد بود.

در نگاه نخست، این رویکرد ممکن است بسیار ناکارآمد به نظر برسد، چرا که مستلزم ذخیرهٔ کل وضعیت شبکه در هر بلاک است. اما در عمل، کارایی آن قابل‌مقایسه با بیت‌کوین است. دلیل این امر آن است که وضعیت شبکه در قالب یک ساختار درختی ذخیره می‌شود و پس از هر بلاک، تنها بخش کوچکی از این درخت نیاز به تغییر دارد. بنابراین، در حالت کلی، میان دو بلاک متوالی، بخش عمده‌ای از درخت

بدون تغییر باقی می‌ماند و داده‌ها می‌توانند تنها یک‌بار ذخیره شده و با استفاده از اشاره‌گرها – یعنی هش زیردرخت‌ها – چندین‌بار مورد ارجاع قرار گیرند.

برای پیاده‌سازی این سازوکار، از نوع خاصی از درخت به نام «درخت پاتریشیا» (Patricia Tree) استفاده می‌شود؛ ساختاری که با اصلاح مفهوم درخت مرکل، امکان درج و حذف گره‌ها – و نه فقط تغییر آن‌ها – را به‌صورت کارآمد فراهم می‌کند. افزون بر این، از آنجا که تمام اطلاعات وضعیت شبکه در آخرین بلاک موجود است، نیازی به ذخیرهٔ کل تاریخچهٔ بلاکچین وجود ندارد؛ راهبردی که اگر در بیت‌کوین قابل استفاده بود، می‌توانست صرفه‌جویی فضایی در حدود ۵ تا ۲۰ برابر ایجاد کند.

کاربردها (اپلیکیشن ها)

به‌طور کلی، سه دستهٔ اصلی از کاربردها بر بستر اتریوم وجود دارد. دستهٔ نخست، کاربردهای مالی هستند که ابزارهای قدرتمندتری برای مدیریت سرمایه و انعقاد قراردادهای مالی در اختیار کاربران قرار می‌دهند. این دسته شامل ارزهای فرعی، مشتقات مالی، قراردادهای پوشش ریسک، کیف‌پول‌های پس‌انداز، وصیت‌نامه‌ها و در نهایت حتی برخی انواع قراردادهای استخدامی در مقیاس کامل می‌شود.

دستهٔ دوم، کاربردهای نیمه‌مالی هستند؛ کاربردهایی که در آن‌ها پول نقش دارد، اما بخش غیرمالی نیز سهم قابل‌توجهی از منطق سیستم را تشکیل می‌دهد. نمونهٔ شاخص این دسته، پاداش‌های خوداجرا برای حل مسائل محاسباتی است.

در نهایت، دستهٔ سوم شامل کاربردهایی می‌شود که اساساً ماهیت مالی ندارند؛ مانند رأی‌گیری آنلاین و سامانه‌های حکمرانی غیرمتمرکز.

سیستم‌های توکن

سیستم‌های توکن روی بلاکچین کاربردهای گسترده‌ای دارند؛ از ارزهای فرعی که نمایانگر دارایی‌هایی مانند دلار آمریکا یا طلا هستند، تا سهام شرکت‌ها، توکن‌های منفرد نمایانگر مالکیت هوشمند، کوپن‌های امن و غیرقابل‌جعل، و حتی سیستم‌های توکنی که هیچ ارتباطی با ارزش‌های متعارف ندارند و صرفاً به‌عنوان سیستم‌های امتیازدهی برای ایجاد انگیزه استفاده می‌شوند.

پیاده‌سازی سیستم‌های توکن در اتریوم به‌طور شگفت‌آوری ساده است. نکتهٔ کلیدی که باید درک شود این است که یک ارز – یا به‌طور کلی یک سیستم توکن – در بنیاد خود چیزی جز یک پایگاه داده با یک عملیات اصلی نیست: کم‌کردن X واحد از A و افزودن X واحد به B ، مشروط بر این‌که (۱) حساب A پیش از تراکنش حداقل X واحد موجودی داشته باشد و (۲) تراکنش توسط A تأیید شده باشد. برای پیاده‌سازی یک سیستم توکن، تنها کافی است این منطق در قالب یک قرارداد پیاده‌سازی شود.

کد پایه برای پیاده‌سازی یک سیستم توکن در زبان Serpent به‌شکل زیر است:

```
from = msg.sender

to = msg.data[0]

value = msg.data[1]


if contract.storage[from] >= value:

    contract.storage[from] = contract.storage[from] - value

    contract.storage[to] = contract.storage[to] + value
```

این کد در اصل پیاده‌سازی مستقیم همان تابع انتقال وضعیت «سیستم بانکی» است که پیش‌تر در این سند توضیح داده شد. برای کامل‌تر شدن سیستم، تنها به چند خط کد اضافه نیاز است؛ از جمله برای توزیع اولیهٔ واحدهای پولی، رسیدگی به برخی حالت‌های خاص، و در حالت ایده‌آل، افزودن تابعی که به سایر قراردادها اجازه دهد موجودی یک آدرس را پرس‌وجو کنند. غیر از این موارد، چیز دیگری لازم نیست.

از نظر تئوریک، سیستم‌های توکن مبتنی بر اتریوم که به‌عنوان ارزهای فرعی عمل می‌کنند، می‌توانند یک قابلیت مهم را داشته باشند که در متاارزهای مبتنی بر بیت‌کوین و روی زنجیره وجود ندارد: امکان پرداخت مستقیم کارمزد تراکنش‌ها با همان ارز. پیاده‌سازی این قابلیت به این صورت است که قرارداد، موجودی‌ای از اتر نگه می‌دارد تا اتر مصرف‌شده برای پرداخت کارمزد را به فرستنده بازگرداند، و این موجودی را از طریق جمع‌آوری واحدهای ارز داخلی به‌عنوان کارمزد و فروش مجدد آن‌ها در یک حراج پیوسته و دائمی، دوباره تأمین می‌کند. به این ترتیب، کاربران لازم است حساب خود را یک‌بار با اتر «فعال» کنند، اما پس از آن، این اتر قابل استفادهٔ مجدد خواهد بود؛ زیرا قرارداد در هر بار استفاده، آن را بازپرداخت می‌کند.

مشتقات مالی و ارزهای با ارزش پایدار

مشتقات مالی رایج‌ترین کاربرد «قراردادهای هوشمند (Smart Contracts) «هستند و در عین حال، از ساده‌ترین موارد برای پیاده‌سازی در قالب کد به‌شمار می‌روند. چالش اصلی در پیاده‌سازی قراردادهای مالی این است که بیشتر آن‌ها نیازمند ارجاع به یک منبع قیمت خارجی هستند. برای مثال، یکی از کاربردهای بسیار مطلوب، قراردادی هوشمند است که در برابر نوسانات قیمت اتر (یا یک رمزارز دیگر) نسبت به دلار آمریکا پوشش ریسک ایجاد کند؛ اما انجام این کار مستلزم آن است که قرارداد از نرخ ETH/USD آگاه باشد.

ساده‌ترین راه برای دستیابی به این اطلاعات، استفاده از یک قرارداد «فید داده (Data Feed) «است که توسط یک نهاد مشخص — برای مثال نزدک — (NASDAQ) نگهداری می‌شود. این قرارداد به‌گونه‌ای طراحی می‌شود که آن نهاد بتواند در صورت نیاز داده‌ها را به‌روزرسانی کند و در عین حال، یک رابط در اختیار سایر قراردادها قرار دهد تا بتوانند با ارسال پیام، پاسخ حاوی قیمت را دریافت کنند.

با در اختیار داشتن این جزء کلیدی، یک قرارداد پوشش ریسک می‌تواند به‌شکل زیر عمل کند:

انتظار برای آن‌که طرف A مقدار ۱۰۰۰ اتر واریز کند.

انتظار برای آن‌که طرف B نیز مقدار ۱۰۰۰ اتر واریز کند.

ثبت ارزش دلاری ۱۰۰۰ اتر — که از طریق پرس‌وجو از قرارداد فید داده محاسبه می‌شود — در فضای ذخیره‌سازی قرارداد؛ فرض می‌کنیم این مقدار برابر با x دلار باشد.

پس از گذشت ۳۰ روز، به A یا B اجازه داده می‌شود با «پینگ» کردن قرارداد، مقدار x دلار اتر (که با مراجعهٔ دوباره به قرارداد فید داده و با قیمت جدید محاسبه می‌شود) به A ارسال شود و باقی‌مانده به B تعلق گیرد.

چنین قراردادی می‌تواند ظرفیت بالایی برای استفاده در تجارت مبتنی بر رمزارز داشته باشد. یکی از اصلی‌ترین انتقادهایی که به رمزارزها وارد می‌شود، نوسان شدید قیمت آن‌هاست. با وجود آن‌که بسیاری از کاربران و پذیرندگان ممکن است امنیت و سهولت استفاده از دارایی‌های رمزنگاری‌شده را بخواهند، اما تمایلی ندارند با خطر از دست دادن ۲۳٪ از ارزش دارایی خود در یک روز مواجه شوند.

تا امروز، رایج‌ترین راه‌حل پیشنهادی برای این مشکل، استفاده از دارایی‌های پشتیبانی‌شده توسط صادرکننده بوده است. در این رویکرد، یک صادرکننده، ارز فرعی‌ای ایجاد می‌کند که اختیار صدور و ابطال واحدهای آن را در دست دارد و در ازای دریافت (خارج از زنجیره) یک واحد از دارایی پایهٔ مشخص — مانند طلا یا دلار آمریکا — یک واحد از آن ارز دیجیتال را به کاربر می‌دهد. سپس صادرکننده متعهد می‌شود که هر زمان یک واحد از دارایی رمزنگاری‌شده به او بازگردانده شود، یک واحد از دارایی پایه را تحویل دهد. این سازوکار امکان «ارتقای» هر دارایی غیررمزنگاری‌شده به یک دارایی رمزنگاری‌شده را فراهم می‌کند؛ مشروط بر آن‌که صادرکننده قابل اعتماد باشد.

اما در عمل، صادرکنندگان همیشه قابل اعتماد نیستند و در برخی کشورها نیز زیرساخت بانکی آن‌قدر ضعیف یا خصمانه است که چنین خدماتی اساساً امکان‌پذیر نیست. مشتقات مالی یک جایگزین برای این رویکرد ارائه می‌دهند. در این مدل، به‌جای آن‌که یک صادرکنندهٔ واحد سرمایهٔ پشتیبان را تأمین کند، یک بازار غیرمتمرکز از سفته‌بازان — که بر افزایش قیمت دارایی مرجع رمزنگاری‌شده شرط می‌بندند — این نقش را بر عهده می‌گیرد.

برخلاف صادرکنندگان، سفته‌بازان امکان خلف وعده ندارند؛ زیرا قرارداد پوشش ریسک، سرمایهٔ آن‌ها را به‌صورت امانی (Escrow) نگه می‌دارد. شایان توجه است که این رویکرد هنوز به‌طور کامل غیرمتمرکز نیست، چرا که همچنان به یک منبع مورد اعتماد برای ارائهٔ داده‌های قیمتی نیاز دارد. با این حال، از نظر کاهش نیاز به زیرساخت‌های پیچیده (برخلاف نقش صادرکننده، ارائهٔ فید قیمت نیاز به مجوز ندارد و احتمالاً می‌توان آن را مصداقی از آزادی بیان دانست) و نیز کاهش امکان تقلب، پیشرفتی بسیار چشمگیر محسوب می‌شود.

سیستم‌های هویت و اعتبار (Identity and Reputation Systems)

قدیمی‌ترین رمزارز جایگزین، یعنیNamecoin ، تلاش داشت با استفاده از یک بلاکچین مشابه بیت‌کوین، یک سیستم ثبت نام ایجاد کند؛ سیستمی که در آن کاربران بتوانند نام‌های خود را به‌همراه داده‌های دیگر در یک پایگاه دادهٔ عمومی ثبت کنند. مهم‌ترین کاربردی که برای این سیستم مطرح شده است، استفاده به‌عنوان یک سامانهٔ DNS است؛ به‌طوری که نام‌های دامنه‌ای مانند) «bitcoin.org» یا در مورد Namecoin، («bitcoin.bit»به یک آدرس IP نگاشت شوند. کاربردهای دیگر شامل احراز هویت ایمیل و در آینده، سیستم‌های اعتبارسنجی پیشرفته‌تر نیز می‌شود.

در ادامه، یک قرارداد پایه برای پیاده‌سازی یک سیستم ثبت نام مشابه Namecoin روی اتریوم آورده شده است:

```
if !contract.storage[tx.data[0]]:
    contract.storage[tx.data[0]] = tx.data[1]
```

این قرارداد بسیار ساده است؛ در واقع چیزی جز یک پایگاه داده درون شبکهٔ اتریوم نیست که می‌توان به آن داده اضافه کرد، اما امکان ویرایش یا حذف داده‌ها از آن وجود ندارد. هر فردی می‌تواند یک نام را همراه با یک مقدار ثبت کند و این ثبت برای همیشه باقی خواهد ماند.

یک قرارداد ثبت نام پیشرفته‌تر می‌تواند شامل یک «بخش تابعی (Function Clause) «نیز باشد که به سایر قراردادها اجازه دهد اطلاعات آن را پرس‌وجو کنند، و همچنین سازوکاری داشته باشد که به «مالک» نام — یعنی نخستین فردی که آن را ثبت کرده است — اجازه دهد دادهٔ مرتبط را تغییر دهد یا مالکیت نام را منتقل کند. حتی می‌توان قابلیت‌های اعتبارسنجی و شبکهٔ اعتماد (Web of Trust) را نیز بر روی چنین سیستمی پیاده‌سازی کرد.

ذخیره‌سازی فایل غیرمتمرکز

در چند سال گذشته، تعدادی از استارتاپ‌های محبوب ذخیره‌سازی آنلاین فایل ظهور کرده‌اند که شاخص‌ترین آن‌ها Dropbox است. هدف این خدمات آن است که کاربران بتوانند یک نسخهٔ پشتیبان از هارددیسک خود بارگذاری کنند و در ازای پرداخت هزینهٔ ماهانه، به داده‌های خود دسترسی داشته باشند. با این حال، بازار ذخیره‌سازی فایل در وضعیت کنونی، در بسیاری موارد ناکارآمد است. بررسی اجمالی راهکارهای موجود نشان می‌دهد که به‌ویژه در بازهٔ موسوم به «درهٔ وهم‌آلود (Uncanny Valley) «یعنی سطح ۲۰ تا ۲۰۰ گیگابایت — جایی که نه سهمیه‌های رایگان اعمال می‌شوند و نه تخفیف‌های سازمانی — هزینهٔ ماهانهٔ ذخیره‌سازی به‌قدری بالاست که گاهی در یک ماه، بیش از قیمت کل یک هارددیسک پرداخت می‌شود.

قراردادهای اتریوم می‌توانند امکان ایجاد یک اکوسیستم ذخیره‌سازی فایل غیرمتمرکز را فراهم کنند؛ اکوسیستمی که در آن کاربران منفرد بتوانند با اجاره‌دادن فضای خالی هارددیسک خود، مقادیر اندکی درآمد کسب کنند و از این طریق، استفاده از فضای بلااستفاده باعث کاهش چشمگیر هزینهٔ ذخیره‌سازی فایل شود.

هستهٔ اصلی چنین سیستمی، چیزی است که آن را «قرارداد دراپ‌باکس غیرمتمرکز (Decentralized Dropbox Contract) «می‌نامیم. این قرارداد به‌صورت زیر عمل می‌کند: ابتدا دادهٔ موردنظر به چندین بلاک تقسیم می‌شود، هر بلاک برای حفظ حریم خصوصی رمزنگاری می‌گردد و سپس یک درخت مرکل از این بلاک‌ها ساخته می‌شود. پس از آن، قراردادی ایجاد می‌شود که طبق آن، هر N بلاک یک‌بار، قرارداد یک اندیس تصادفی در درخت مرکل انتخاب می‌کند (با استفاده از هش بلاک قبلی – که از داخل کد قرارداد قابل دسترسی است – به‌عنوان منبع تصادفی) و مقدار X اتر به نخستین موجودیتی پرداخت می‌کند که تراکنشی شامل یک اثبات مالکیت شبیه به SPV برای بلاک مربوط به آن اندیس خاص ارائه دهد.

هنگامی که کاربر بخواهد فایل خود را دوباره دانلود کند، می‌تواند از یک پروتکل کانال پرداخت خرد (Micropayment Channel) استفاده کند؛ برای مثال، پرداخت ۱ سابو (szabo) به‌ازای هر ۳۲ کیلوبایت. کارآمدترین روش از نظر کارمزد آن است که پرداخت‌کننده تا پایان فرآیند، تراکنش را منتشر نکند و به‌جای آن، پس از هر ۳۲ کیلوبایت، تراکنش قبلی را با یک تراکنش اندکی سودآورتر با همان نانس جایگزین کند.

یکی از ویژگی‌های مهم این پروتکل آن است که اگرچه در نگاه نخست ممکن است به نظر برسد کاربر به تعداد زیادی گرهٔ تصادفی اعتماد می‌کند که فایل را حذف نکنند، اما می‌توان این ریسک را تقریباً به صفر رساند. این کار با تقسیم فایل به بخش‌های متعدد از طریق اشتراک‌گذاری راز (Secret Sharing) و نظارت بر قراردادها برای اطمینان از این‌که هر بخش همچنان در اختیار حداقل یک گره قرار دارد انجام می‌شود. تا زمانی که یک قرارداد همچنان در حال پرداخت پاداش باشد، این موضوع یک اثبات رمزنگاری‌شده فراهم می‌کند که نشان می‌دهد هنوز فردی در شبکه، فایل را ذخیره کرده است.

سازمان‌های خودگردان غیرمتمرکز

مفهوم کلی یک «سازمان غیرمتمرکز (Decentralized Organization) «به یک نهاد مجازی اشاره دارد که مجموعه‌ای از اعضا یا سهام‌داران دارد؛ افرادی که معمولاً با اکثریتی مانند ۶۷٪، اختیار خرج‌کردن سرمایه‌های سازمان و تغییر کد آن را در اختیار دارند. اعضا به‌صورت جمعی تصمیم می‌گیرند که منابع مالی سازمان چگونه تخصیص داده شوند. روش‌های تخصیص سرمایه در یک سازمان غیرمتمرکز می‌تواند از پرداخت پاداش‌ها و حقوق‌ها گرفته تا سازوکارهای پیچیده‌تری مانند ایجاد یک ارز داخلی برای پاداش‌دادن به فعالیت‌ها را شامل شود. این ساختار، در عمل چارچوب‌های حقوقی یک شرکت یا سازمان غیرانتفاعی سنتی را بازتولید می‌کند، با این تفاوت که تمام سازوکارهای اجرایی آن صرفاً بر پایهٔ فناوری رمزنگاری و بلاکچین پیاده‌سازی می‌شوند.

تا امروز، بخش عمده‌ای از بحث‌ها پیرامونDAO ها بر مدل «سرمایه‌داری» یک «شرکت خودگردان غیرمتمرکز Decentralized) « Autonomous Corporationیا (DAC متمرکز بوده است؛ مدلی که در آن سهام‌داران سود سهام دریافت می‌کنند و سهام قابل معامله است. در مقابل، یک مدل جایگزین که می‌توان آن را «جامعهٔ خودگردان غیرمتمرکز» نامید، تمام اعضا را در تصمیم‌گیری دارای سهم برابر می‌داند و برای افزودن یا حذف یک عضو، به موافقت ۶۷٪ اعضای موجود نیاز دارد. در چنین مدلی، شرط «هر فرد فقط یک عضویت» باید به‌صورت جمعی توسط خود اعضا اعمال شود.

یک چارچوب کلی برای پیاده‌سازی کد یک سازمان غیرمتمرکز به این صورت است: ساده‌ترین طراحی، مجموعه‌ای از کدهای خوداصلاح‌شونده است که در صورتی تغییر می‌کنند که دو سوم اعضا با تغییر موردنظر موافقت کنند. هرچند کدها از نظر نظری غیرقابل‌تغییر هستند، اما می‌توان با قراردادن بخش‌های مختلف کد در قراردادهای جداگانه و ذخیرهٔ آدرس قراردادهای فراخوانی‌شده در فضای ذخیره‌سازی قابل‌تغییر، به‌طور عملی به نوعی تغییرپذیری دست یافت.

در یک پیاده‌سازی ساده از چنین قراردادDAO ، سه نوع تراکنش وجود دارد که بر اساس دادهٔ همراه تراکنش از یکدیگر متمایز می‌شوند:

[0, i, K, V]برای ثبت یک پیشنهاد با شناسهٔ i به‌منظور تغییر مقدار ذخیره‌شده در اندیس K به مقدارV

[1, i]برای ثبت رأی موافق با پیشنهاد i

[2, i]برای نهایی‌سازی پیشنهاد i در صورتی که تعداد رأی‌ها به حد نصاب رسیده باشد

قرارداد، برای هر یک از این موارد منطق جداگانه‌ای خواهد داشت. همچنین، فهرستی از تمام تغییرات باز در فضای ذخیره‌سازی به‌همراه لیست رأی‌دهندگان هر تغییر نگه‌داری می‌شود. علاوه بر این، قرارداد دارای فهرستی از تمام اعضا است. هنگامی که هر تغییر ذخیره‌سازی به رأی موافق دو سوم اعضا برسد، یک تراکنش نهایی‌سازی می‌تواند آن تغییر را اجرا کند.

یک چارچوب پیشرفته‌تر می‌تواند قابلیت رأی‌گیری داخلی برای عملیات‌هایی مانند ارسال تراکنش، افزودن عضو جدید یا حذف اعضا را نیز در خود داشته باشد، و حتی از تفویض رأی به سبک «دموکراسی سیال (Liquid Democracy) «پشتیبانی کند؛ به این معنا که هر فرد می‌تواند رأی خود را به فرد دیگری واگذار کند، و این واگذاری به‌صورت زنجیره‌ای منتقل شود (اگر A رأی خود را به B بدهد و B آن را به C واگذار کند، رأی A توسط C تعیین می‌شود). چنین طراحی‌ای به سازمان غیرمتمرکز اجازه می‌دهد به‌صورت ارگانیک رشد کند و به‌تدریج وظیفهٔ پالایش عضویت به متخصصان سپرده شود؛ با این تفاوت که برخلاف «سیستم‌های سنتی»، این متخصصان می‌توانند به‌راحتی و در طول زمان وارد یا خارج شوند، چرا که اعضای جامعه همواره در حال تغییر هم‌سویی‌های خود هستند.

مدل جایگزین دیگر، یک شرکت غیرمتمرکز است که در آن هر حساب می‌تواند صفر یا چند سهم داشته باشد و برای تصمیم‌گیری، موافقت دارندگان دو سوم سهام لازم است. یک چارچوب کامل برای چنین مدلی شامل قابلیت مدیریت دارایی‌ها، امکان ارائهٔ پیشنهاد خرید یا فروش سهام، و امکان پذیرش این پیشنهادها — ترجیحاً با یک سازوکار تطبیق سفارش درون قرارداد — خواهد بود. در این مدل نیز تفویض رأی به سبک دموکراسی سیال وجود دارد و مفهوم «هیئت‌مدیره» را تعمیم می‌دهد.

در آینده، ممکن است سازوکارهای پیشرفته‌تری برای حکمرانی سازمانی پیاده‌سازی شوند؛ در این نقطه است که یک سازمان غیرمتمرکز (DO)می‌تواند به‌درستی «سازمان خودگردان غیرمتمرکز (DAO) «نامیده شود. مرز میان DO و DAO دقیق و شفاف نیست، اما معیار کلی تمایز آن‌ها این است که آیا حکمرانی عمدتاً از طریق فرآیندهایی شبیه به سیاست انجام می‌شود یا به‌صورت «خودکار». یک آزمون شهودی مناسب، معیار «نبود زبان مشترک» است: آیا سازمان می‌تواند همچنان به فعالیت خود ادامه دهد اگر هیچ دو عضوی زبان مشترکی نداشته باشند؟ روشن است که یک شرکت سنتی با ساختار سهام‌داری در چنین شرایطی شکست می‌خورد، در حالی که چیزی مانند پروتکل بیت‌کوین شانس بسیار بیشتری برای موفقیت دارد.

نظریهٔ فوتارکی (Futarchy) رابین هنسن — که سازوکاری برای حکمرانی سازمانی از طریق بازارهای پیش‌بینی است — نمونهٔ خوبی از آن چیزی است که حکمرانی واقعاً «خودگردان» می‌تواند باشد. با این حال، نباید فرض کرد که همهٔ DAOها الزاماً از همهٔ DOها بهتر هستند؛ خودکارسازی صرفاً یک الگو است که در برخی حوزه‌ها مزایای بسیار بزرگی دارد و در برخی دیگر ممکن است عملی نباشد. به همین دلیل، وجود تعداد زیادی سازمان نیمه‌خودگردان (Semi-DAO) نیز کاملاً محتمل است.

کاربردهای بیشتر (Further Applications)

کیف‌پول‌های پس‌انداز

فرض کنید آلیس می‌خواهد دارایی‌های خود را در امنیت نگه دارد، اما نگران است که کلید خصوصی‌اش را گم کند یا مورد نفوذ قرار بگیرد. او مقداری اتر را در قراردادی به‌همراه باب — که می‌تواند یک بانک باشد — قرار می‌دهد، با قوانین زیر:

آلیس به‌تنهایی می‌تواند حداکثر ۱٪ از موجودی را در هر روز برداشت کند.

باب نیز به‌تنهایی می‌تواند حداکثر ۱٪ از موجودی را در هر روز برداشت کند، اما آلیس این امکان را دارد که با یک تراکنش، این اختیار را از باب سلب کند.

آلیس و باب با هم می‌توانند هر مقدار دلخواهی را برداشت کنند.

در حالت عادی، برداشت روزانهٔ ۱٪ برای آلیس کافی است و اگر بخواهد مبلغ بیشتری برداشت کند، می‌تواند از باب کمک بگیرد. اگر کلید آلیس هک شود، او می‌تواند سریعاً به باب مراجعه کند تا دارایی‌ها به یک قرارداد جدید منتقل شوند. اگر آلیس کلید خود را گم کند، باب در نهایت قادر خواهد بود دارایی‌ها را خارج کند. و اگر باب رفتاری مخرب داشته باشد، آلیس می‌تواند اختیار برداشت او را غیرفعال کند.

بیمهٔ محصولات کشاورزی

می‌توان به‌راحتی یک قرارداد مشتقهٔ مالی ایجاد کرد که به‌جای شاخص قیمت، از یک دادهٔ آب‌وهوایی استفاده کند. اگر یک کشاورز در آیووا قراردادی خریداری کند که پرداخت آن به‌صورت معکوس به میزان بارندگی وابسته باشد، در صورت خشکسالی به‌طور خودکار پول دریافت می‌کند و در صورت بارندگی کافی، از عملکرد خوب محصولاتش سود خواهد برد.

فید دادهٔ غیرمتمرکز

برای قراردادهای مالی مبتنی بر اختلاف قیمت، می‌توان فید داده را با استفاده از پروتکلی به نام شلینگ‌کوین (SchellingCoin) غیرمتمرکز کرد. در این روش، Nشرکت‌کننده مقدار یک دادهٔ مشخص (مثلاً قیمت ETH/USD) را ارائه می‌دهند. مقادیر مرتب می‌شوند و افرادی که مقدارشان بین صدک ۲۵ تا ۷۵ قرار دارد، یک توکن پاداش می‌گیرند. از آنجا که همه انگیزه دارند مقداری را اعلام کنند که دیگران نیز اعلام می‌کنند، تنها مقداری که جمع بزرگی از افراد می‌توانند بر آن توافق کنند، مقدار بدیهی یعنی «حقیقت» است. این سازوکار می‌تواند برای ارائهٔ داده‌هایی مانند قیمتETH/USD ، دمای هوا در برلین یا حتی نتیجهٔ یک محاسبهٔ سنگین استفاده شود.

امانت‌داری چندامضایی هوشمند

بیت‌کوین امکان قراردادهای چندامضایی را فراهم می‌کند؛ برای مثال، سه امضا از پنج کلید برای خرج‌کردن دارایی کافی است. اتریوم این مفهوم را با جزئیات بسیار بیشتری پیاده‌سازی می‌کند؛ برای نمونه:

چهار امضا از پنج می‌توانند کل موجودی را برداشت کنند،

سه امضا از پنج می‌توانند روزانه تا ۱۰٪ برداشت کنند،

دو امضا از پنج می‌توانند روزانه تا ۰٫۵٪ برداشت کنند.

علاوه بر این، چندامضایی در اتریوم غیرهم‌زمان است؛ یعنی امضاها می‌توانند در زمان‌های متفاوت ثبت شوند و با ثبت آخرین امضا، تراکنش به‌صورت خودکار انجام می‌شود.

محاسبات ابری

فناوری EVM می‌تواند برای ایجاد یک محیط محاسباتی قابل‌راستی‌آزمایی استفاده شود؛ به‌طوری که کاربران از دیگران بخواهند محاسباتی را انجام دهند و سپس برای برخی نقاط تصادفی، اثبات صحت محاسبه ارائه شود. این امکان ایجاد بازاری برای محاسبات ابری را فراهم می‌کند که در آن هر کاربری با رایانهٔ شخصی، لپ‌تاپ یا سرور تخصصی می‌تواند مشارکت کند. بررسی‌های تصادفی به‌همراه سپرده‌های امنیتی تضمین می‌کنند که تقلب سودآور نباشد. البته این مدل برای همهٔ انواع محاسبات مناسب نیست؛ به‌ویژه کارهایی که به ارتباط شدید بین فرایندها نیاز دارند. اما بسیاری از وظایف به‌راحتی قابل موازی‌سازی هستند؛ مانند پروژه‌هایی نظیرSETI@home ، folding@homeو الگوریتم‌های ژنتیک.

قمار همتابه‌همتا

پروتکل‌های متعددی برای قمار همتابه‌همتا، مانندCyberdice ، می‌توانند روی بلاکچین اتریوم پیاده‌سازی شوند. ساده‌ترین شکل قمار در واقع یک قرارداد مبتنی بر اختلاف روی هش بلاک بعدی است، و پروتکل‌های پیچیده‌تر را می‌توان بر پایهٔ آن ساخت؛ به‌گونه‌ای که خدمات قمار با کارمزد نزدیک به صفر و بدون امکان تقلب ارائه شوند.

بازارهای پیش‌بینی

با وجود یک اوراکل یا شلینگ‌کوین، پیاده‌سازی بازارهای پیش‌بینی نیز بسیار ساده است. ترکیب بازارهای پیش‌بینی با شلینگ‌کوین می‌تواند نخستین کاربرد فراگیر فوتارشی (Futarchy) به‌عنوان یک پروتکل حاکمیتی برای سازمان‌های غیرمتمرکز باشد.

بازارهای غیرمتمرکز درون‌زنجیره‌ای

با استفاده از سیستم‌های هویت و اعتبار به‌عنوان زیرساخت، می‌توان بازارهای غیرمتمرکز کاملاً درون‌زنجیره‌ای ایجاد کرد.

متفرقه‌ها و ملاحظات

پیاده‌سازی اصلاح‌شدهٔGHOST

پروتکل «Greedy Heaviest Observed Subtree» یا به‌اختصارGHOST ، نوآوری‌ای است که نخستین‌بار در دسامبر ۲۰۱۳ توسط یوناتان سومپولینسکی (Yonatan Sompolinsky) و آویو زوهار (Aviv Zohar) معرفی شد. انگیزهٔ اصلی پشت GHOST این واقعیت است که بلاکچین‌هایی با زمان تأیید سریع، در وضعیت فعلی با کاهش امنیت مواجه می‌شوند؛ چرا که نرخ بالای بلاک‌های منقضی‌شده یا بلاک‌های کهنه (stale blocks) دارند. علت این مسئله آن است که انتشار یک بلاک در شبکه به زمان نیاز دارد؛ اگر استخراج‌کنندهٔ A یک بلاک استخراج کند و پیش از آن‌که این بلاک به استخراج‌کنندهٔ B برسد، استخراج‌کنندهٔ B نیز بلاک دیگری استخراج کند، بلاک B در نهایت بلاک منقضی‌شده محسوب می‌شود و هیچ نقشی در امنیت شبکه نخواهد داشت.

علاوه بر این، یک مشکل تمرکزگرایی نیز وجود دارد. اگر استخراج‌کنندهٔ A یک استخر استخراج با ۳۰٪ توان هش (hashpower) باشد و استخراج‌کنندهٔ B تنها ۱۰٪ توان هش داشته باشد، احتمال تولید بلاک منقضی‌شده برای A حدود ۷۰٪ و برای B حدود ۹۰٪ خواهد بود. دلیل این تفاوت آن است که در ۳۰٪ مواقع، خود A بلاک قبلی را استخراج کرده و بلافاصله به داده‌های استخراج دسترسی دارد. در نتیجه، اگر فاصلهٔ زمانی بین بلاک‌ها آن‌قدر کوتاه باشد که نرخ بلاک‌های منقضی‌شده بالا برود، استخراج‌کنندهٔ بزرگ‌تر صرفاً به‌دلیل اندازهٔ خود کارایی بسیار بالاتری خواهد داشت. ترکیب این دو عامل باعث می‌شود بلاکچین‌هایی که بلاک‌ها را با سرعت بالا تولید می‌کنند، به احتمال زیاد به تمرکز قدرت استخراج در یک استخر بزرگ منجر شوند؛ استخری که عملاً کنترل فرآیند استخراج را در دست می‌گیرد.

مطابق توضیح سومپولینسکی و زوهار، پروتکل GHOST مشکل نخست یعنی کاهش امنیت شبکه را با واردکردن بلاک‌های منقضی‌شده در محاسبهٔ «طولانی‌ترین زنجیره» حل می‌کند. به این معنا که نه‌تنها بلاک والد و نیاکان آن، بلکه فرزندان منقضی‌شدهٔ نیاکان بلاک نیز — که در اصطلاح اتریوم «عموها (uncles) «نامیده می‌شوند — در محاسبهٔ مجموع اثبات کار پشتیبان هر زنجیره لحاظ می‌شوند.

برای حل مشکل دوم، یعنی سوگیری تمرکزگرایانه، اتریوم فراتر از نسخهٔ اولیهٔ GHOST حرکت می‌کند و اجازه می‌دهد بلاک‌های منقضی‌شده در زنجیرهٔ اصلی ثبت شوند و پاداش دریافت کنند. در این مدل، یک بلاک منقضی‌شده ۹۳٫۷۵٪ از پاداش پایهٔ بلاک را دریافت می‌کند و بلاکی که آن را در زنجیره قرار می‌دهد (اصطلاحاً «برادرزاده» یا(nephew ، ٪۶٫۲۵ باقی‌مانده را دریافت می‌کند. با این حال، کارمزد تراکنش‌ها به بلاک‌های عمو تعلق نمی‌گیرد.

اتریوم یک نسخهٔ ساده‌شده از GHOST را پیاده‌سازی می‌کند که تنها تا پنج سطح عمق دارد. به‌طور مشخص، یک بلاک منقضی‌شده فقط می‌تواند توسط فرزندان نسل دوم تا پنجم بلاک والد خود به‌عنوان عمو در زنجیره قرار گیرد و بلاک‌هایی با فاصلهٔ نسبی بیشتر (برای مثال فرزند نسل ششم والد یا فرزند نسل سوم پدربزرگ) مجاز نیستند.

این محدودیت به چند دلیل اعمال شده است. نخست آن‌که GHOST بدون محدودیت، محاسبهٔ معتبر بودن عموها برای هر بلاک را بسیار پیچیده می‌کند. دوم این‌که GHOST نامحدود همراه با پرداخت پاداش — آن‌گونه که در اتریوم استفاده می‌شود — انگیزهٔ استخراج روی زنجیرهٔ اصلی را کاهش داده و ممکن است استخراج‌کنندگان را به استخراج روی زنجیرهٔ مهاجمان عمومی سوق دهد. در نهایت، محاسبات نشان می‌دهد که GHOST پنج‌سطحی همراه با مشوق‌های مالی، حتی با زمان بلاک ۱۵ ثانیه، بیش از ۹۵٪ کارایی دارد و استخراج‌کنندگانی با ۲۵٪ توان هش، افزایش تمرکز کمتر از ۳٪ را تجربه می‌کنند.

کارمزدها(Fees)

از آنجا که هر تراکنشی که در بلاکچین منتشر می‌شود، هزینه‌ای را به شبکه تحمیل می‌کند — زیرا همهٔ گره‌ها باید آن را دانلود و اعتبارسنجی کنند — وجود یک سازوکار تنظیم‌کننده برای جلوگیری از سوءاستفاده ضروری است؛ سازوکاری که معمولاً بر پایهٔ کارمزد تراکنش‌ها عمل می‌کند. رویکرد پیش‌فرض در بیت‌کوین، استفاده از کارمزدهای کاملاً داوطلبانه است و این وظیفه را به استخراج‌کنندگان واگذار می‌کند که به‌عنوان دروازه‌بان عمل کرده و حداقل‌های پویا تعیین کنند. این رویکرد در جامعهٔ بیت‌کوین بسیار مورد استقبال قرار گرفته، به‌ویژه به این دلیل که «بازارمحور (market-based) «است و اجازه می‌دهد عرضه و تقاضا میان استخراج‌کنندگان و ارسال‌کنندگان تراکنش، قیمت را تعیین کند.

اما مشکل این استدلال آن است که پردازش تراکنش در واقع یک بازار نیست. هرچند در نگاه اول جذاب به‌نظر می‌رسد که پردازش تراکنش را خدمتی بدانیم که استخراج‌کننده به ارسال‌کننده ارائه می‌دهد، اما در واقعیت، هر تراکنشی که یک استخراج‌کننده در بلاک قرار می‌دهد باید توسط تمام گره‌های شبکه پردازش شود. در نتیجه، بخش عمدهٔ هزینهٔ پردازش تراکنش بر دوش اشخاص ثالث است، نه استخراج‌کننده‌ای که تصمیم به افزودن تراکنش گرفته است. از این‌رو، بروز پدیدهٔ «تراژدی منابع مشترک (tragedy of the commons) « بسیار محتمل خواهد بود.

با این حال، مشخص می‌شود که این نقص در سازوکار بازارمحور، تحت یک مجموعه فرض ساده‌ساز — هرچند نه‌چندان دقیق — به‌طور شگفت‌انگیزی خودبه‌خود خنثی می‌شود. استدلال به این صورت است. فرض کنیم:

یک تراکنش منجر به اجرای k عملیات شود و پاداشی معادل kR به استخراج‌کننده‌ای که آن را در بلاک قرار دهد بدهد، که در آن R توسط ارسال‌کننده تعیین شده و مقادیر k و) R تقریباً) از پیش برای استخراج‌کننده قابل مشاهده‌اند.

هر عملیات هزینه‌ای معادل C برای هر گره دارد (یعنی تمام گره‌ها بهره‌وری یکسانی دارند).

تعداد N گرهٔ استخراج‌کننده وجود دارد که هرکدام دقیقاً سهم برابری از توان پردازشی دارند (یعنی 1 N/کل توان).

هیچ گرهٔ کاملِ غیر استخراج‌کننده‌ای وجود ندارد.

در این حالت، یک استخراج‌کننده زمانی حاضر به پردازش تراکنش است که پاداش مورد انتظار آن از هزینه بیشتر باشد. پاداش مورد انتظار برابر با kR/N است، زیرا استخراج‌کننده تنها با احتمال 1 N/بلاک بعدی را استخراج می‌کند، و هزینهٔ پردازش برای استخراج‌کننده برابر با kC است. بنابراین، استخراج‌کنندگان تراکنش‌هایی را می‌پذیرند که kR/N > kC باشد، یا به‌عبارت دیگر .R > NC از آنجا که R کارمزد هر عملیات است، این مقدار حداقلی از سودی را نشان می‌دهد که ارسال‌کننده از تراکنش به‌دست می‌آورد، و NC نیز هزینهٔ کل شبکه برای اجرای یک عملیات است. در نتیجه، استخراج‌کنندگان انگیزه دارند تنها تراکنش‌هایی را بپذیرند که سود کلی آن‌ها بیش از هزینهٔ تحمیلی به شبکه باشد.

اما در عمل، این فرض‌ها به‌طور کامل برقرار نیستند:

استخراج‌کننده هزینهٔ بیشتری نسبت به سایر گره‌های اعتبارسنج برای پردازش تراکنش می‌پردازد، زیرا زمان اعتبارسنجی اضافه، انتشار بلاک را به تأخیر می‌اندازد و احتمال منقضی‌شدن بلاک را افزایش می‌دهد.

گره‌های کاملِ غیر استخراج‌کننده واقعاً وجود دارند.

توزیع توان استخراج در عمل می‌تواند به‌شدت نابرابر باشد.

افرادی با انگیزه‌های مخرب — مانند سوداگران، دشمنان سیاسی یا افراد نامتعادل — وجود دارند که مطلوبیت آن‌ها در آسیب‌زدن به شبکه تعریف می‌شود و می‌توانند قراردادهایی طراحی کنند که هزینهٔ آن‌ها برای خودشان کم، اما هزینهٔ تحمیلی به سایر گره‌ها بسیار بالا باشد.

عامل اول باعث می‌شود استخراج‌کنندگان تمایل کمتری به افزودن تراکنش‌ها داشته باشند و عامل دوم مقدار NC را افزایش می‌دهد؛ بنابراین این دو اثر تا حدی یکدیگر را خنثی می‌کنند. اما عوامل سوم و چهارم مسئلهٔ اصلی هستند. برای حل آن‌ها، اتریوم یک سقف شناور اعمال می‌کند: هیچ بلاکی نمی‌تواند بیش از BLK_LIMIT_FACTOR برابر میانگین نمایی متحرک بلندمدت عملیات داشته باشد. به‌طور مشخص:

```
blk.oplimit = floor(
  (blk.parent.oplimit * (EMA_FACTOR - 1) +
   floor(parent.opcount * BLK_LIMIT_FACTOR))
  / EMA_FACTOR
)
```

در حال حاضر، مقادیر BLK_LIMIT_FACTOR و EMA_FACTOR به‌ترتیب 65536 و 1.5 تعیین شده‌اند، اما احتمالاً پس از تحلیل‌های بیشتر تغییر خواهند کرد.

محاسبه و تورینگ‌کامل‌بودن (Computation And Turing-Completeness)

نکتهٔ مهم این است که ماشین مجازی اتریوم (Ethereum Virtual Machine یا EVM) تورینگ‌کامل (Turing-complete) است؛ به این معنا که کدهای EVM می‌توانند هر محاسبه‌ای را که از نظر نظری قابل انجام است پیاده‌سازی کنند، از جمله حلقه‌های بی‌نهایت. کد EVMامکان ایجاد حلقه را به دو روش فراهم می‌کند. نخست، از طریق دستور JUMP که اجازه می‌دهد برنامه به نقطه‌ای قبلی در کد بازگردد، و دستور JUMPI برای پرش شرطی، که امکان ساخت عباراتی مانند while x < 27: x = x * 2 را فراهم می‌کند. دوم، قراردادها می‌توانند قراردادهای دیگر را فراخوانی کنند که این موضوع امکان ایجاد حلقه از طریق بازگشت (recursion) را فراهم می‌سازد.

این قابلیت به‌طور طبیعی یک مشکل ایجاد می‌کند: آیا کاربران مخرب می‌توانند با وادارکردن استخراج‌کنندگان و گره‌های کامل به ورود به یک حلقهٔ بی‌نهایت، عملاً شبکه را از کار بیندازند؟ این مسئله ناشی از مشکلی شناخته‌شده در علوم کامپیوتر است که «مسئلهٔ توقف « (Halting Problem)نام دارد؛ یعنی به‌طور کلی هیچ راهی وجود ندارد که بتوان از پیش تعیین کرد آیا یک برنامه در نهایت متوقف خواهد شد یا خیر.

همان‌طور که در بخش تابع انتقال وضعیت توضیح داده شد، راه‌حل اتریوم این است که هر تراکنش باید حداکثر تعداد گام‌های محاسباتی مجاز خود را مشخص کند. اگر اجرای کد از این حد فراتر رود، محاسبه بازگردانده می‌شود، اما کارمزد همچنان پرداخت خواهد شد. پیام‌ها نیز دقیقاً به همین شیوه عمل می‌کنند. برای درک بهتر انگیزهٔ این راه‌حل، مثال‌های زیر را در نظر بگیرید:

یک مهاجم قراردادی ایجاد می‌کند که یک حلقهٔ بی‌نهایت اجرا می‌کند و سپس تراکنشی برای فعال‌سازی آن به استخراج‌کننده ارسال می‌کند. استخراج‌کننده تراکنش را پردازش کرده و حلقهٔ بی‌نهایت را اجرا می‌کند تا زمانی که گس تمام شود. با وجود آن‌که اجرا در میانهٔ کار متوقف می‌شود، تراکنش همچنان معتبر است و استخراج‌کننده کارمزد هر گام محاسباتی را از مهاجم دریافت می‌کند.

یک مهاجم حلقه‌ای بسیار طولانی ایجاد می‌کند تا استخراج‌کننده را وادار کند آن‌قدر محاسبه انجام دهد که تا پایان اجرا چند بلاک جدید استخراج شود و دیگر امکان گنجاندن تراکنش در بلاک برای دریافت کارمزد وجود نداشته باشد. با این حال، مهاجم مجبور است مقدار STARTGASرا از پیش مشخص کند و این مقدار سقف تعداد گام‌های محاسباتی را تعیین می‌کند؛ بنابراین استخراج‌کننده از ابتدا می‌داند که اجرای این تراکنش چه‌قدر سنگین خواهد بود.

یک مهاجم قراردادی با کدی از این جنس مشاهده می‌کند:

send(A, contract.storage[A]); contract.storage[A] = 0

سپس تراکنشی ارسال می‌کند که فقط به‌اندازهٔ اجرای دستور اول گس دارد، اما برای اجرای دستور دوم کافی نیست (یعنی برداشت انجام می‌شود، اما موجودی صفر نمی‌شود). نویسندهٔ قرارداد نیازی به محافظت خاص در برابر این نوع حمله ندارد، زیرا اگر اجرا در میانهٔ راه متوقف شود، تمام تغییرات بازگردانده می‌شوند.

یک قرارداد مالی برای کاهش ریسک، مقدار میانهٔ ۹ منبع دادهٔ اختصاصی را محاسبه می‌کند. مهاجم کنترل یکی از این منابع داده را که از طریق سازوکار فراخوانی با آدرس متغیر (توضیح‌داده‌شده در بخشDAO ها) قابل تغییر است، به‌دست می‌گیرد و آن را به یک حلقهٔ بی‌نهایت تبدیل می‌کند تا تمام تلاش‌ها برای برداشت وجه از قرارداد مالی با اتمام گس مواجه شود. با این حال، قرارداد مالی می‌تواند برای پیام ارسالی یک محدودیت گس تعیین کند و از این مشکل جلوگیری نماید.

جایگزین تورینگ‌کامل‌بودن، تورینگ‌ناقص‌بودن (Turing-incompleteness) است؛ حالتی که در آن دستورهای JUMP و JUMPI وجود ندارند و در هر لحظه فقط یک نسخه از هر قرارداد می‌تواند در پشتهٔ فراخوانی قرار گیرد. در چنین سیستمی، شاید نیازی به سازوکار کارمزد و عدم‌قطعیت‌های مربوط به اثربخشی راه‌حل فعلی نبود، زیرا هزینهٔ اجرای یک قرارداد به‌طور قطعی توسط اندازهٔ آن محدود می‌شد. افزون بر این، تورینگ‌ناقص‌بودن حتی محدودیت بزرگی هم به‌نظر نمی‌رسد؛ از میان تمام نمونه‌قراردادهایی که تاکنون به‌صورت داخلی بررسی شده‌اند، فقط یکی به حلقه نیاز داشت و حتی آن هم می‌توانست با تکرار ۲۶ بارهٔ یک خط کد جایگزین شود.

با توجه به پیامدهای جدی تورینگ‌کامل‌بودن و مزایای ظاهراً محدود آن، این پرسش مطرح می‌شود: چرا از یک زبان تورینگ‌ناقص استفاده نکنیم؟ اما در عمل، تورینگ‌ناقص‌بودن راه‌حل تمیزی برای این مشکل نیست. برای روشن‌شدن موضوع، قراردادهای زیر را در نظر بگیرید:

C0 → C1 → C2 → ...

C4: call(call(call(C1, C2, C3)));

C9: call(C50); call(C50);

C50: (اجرای یک گام از برنامه و ثبت تغییر در فضای ذخیره‌سازی)

اکنون یک تراکنش به C0 ارسال می‌شود. در نتیجه، با ۵۱ تراکنش، قراردادی خواهیم داشت که ۲۵۰ گام محاسباتی اجرا می‌کند. استخراج‌کنندگان می‌توانند تلاش کنند با نگه‌داشتن مقداری در کنار هر قرارداد — که حداکثر تعداد گام‌های محاسباتی آن را مشخص می‌کند — چنین «بمب‌های منطقی» را از پیش شناسایی کنند و این مقدار را برای فراخوانی‌های بازگشتی محاسبه نمایند. اما این کار

مستلزم آن است که استخراج‌کنندگان قراردادهایی را که قراردادهای دیگر ایجاد می‌کنند ممنوع کنند؛ چرا که ایجاد و اجرای تمام ۵۰ قرارداد بالا می‌تواند به‌راحتی در قالب یک قرارداد واحد انجام شود.

مشکل دیگر این است که فیلد آدرس در یک پیام متغیر است؛ بنابراین، به‌طور کلی حتی ممکن نیست از پیش تعیین کرد که یک قرارداد دقیقاً کدام قراردادهای دیگر را فراخوانی خواهد کرد. در نهایت، به نتیجه‌ای جالب می‌رسیم: مدیریت تورینگ‌کامل‌بودن به‌طور شگفت‌انگیزی ساده است، و در مقابل، نبود تورینگ‌کامل‌بودن به‌طرز غیرمنتظره‌ای دشوار است — مگر آن‌که دقیقاً همان کنترل‌ها اعمال شوند. و در این صورت، چرا اصلاً پروتکل را تورینگ‌کامل نکنیم؟

ارز و سازوکار صدور (Currency And Issuance)

شبکهٔ اتریوم دارای یک ارز درونی اختصاصی به نام اتر (Ether) است که دو نقش اصلی را ایفا می‌کند:

نخست، فراهم‌کردن یک لایهٔ نقدشوندگی پایه برای امکان تبادل کارآمد میان انواع مختلف دارایی‌های دیجیتال، و مهم‌تر از آن، ایجاد یک سازوکار برای پرداخت کارمزد تراکنش‌ها.

برای سهولت استفاده و جلوگیری از اختلافات آینده (مانند بحث فعلی mBTC / uBTC / satoshi در بیت‌کوین)، واحدهای پولی از پیش نام‌گذاری شده‌اند:

وی(wei)

10^{12} : سزابو(szabo)

10^{15} : فینی(finney)

10^{18} : اتر(ether)

این ساختار را می‌توان نسخهٔ گسترش‌یافته‌ای از مفاهیمی مانند «دلار و سنت» یا «بیت‌کوین و ساتوشی» دانست. انتظار می‌رود در آیندهٔ نزدیک،

اتر برای تراکنش‌های عادی،

فینی برای ریزتراکنش‌ها،

و سزابو و وی برای مباحث فنی مرتبط با کارمزدها و پیاده‌سازی پروتکل مورد استفاده قرار گیرند.

مدل صدور ارز به‌صورت زیر خواهد بود:

اتر از طریق یک فروش ارزی (currency sale) با نرخ ۱۳۳۷ تا ۲۰۰۰ اتر به‌ازای هر بیت‌کوین عرضه می‌شود. هدف از این سازوکار، تأمین مالی سازمان اتریوم و پرداخت هزینه‌های توسعه است؛ روشی که پیش‌تر توسط چندین پلتفرم رمزنگاری دیگر با موفقیت استفاده شده است. خریداران اولیه از تخفیف‌های بیشتری بهره‌مند خواهند شد. بیت‌کوین‌های حاصل از این فروش به‌طور کامل برای پرداخت حقوق و پاداش به توسعه‌دهندگان، پژوهشگران و پروژه‌های فعال در اکوسیستم ارزهای رمزنگاری‌شده مصرف خواهد شد.

معادل ۰٫۰۹۹ برابر از کل مقدار فروخته‌شده به مشارکت‌کنندگان اولیه‌ای اختصاص می‌یابد که پیش از فراهم‌شدن تأمین مالی از طریق بیت‌کوین یا حتی اطمینان از وجود آن، در توسعه مشارکت داشته‌اند. همچنین ۰٫۰۹۹ برابر دیگر به پروژه‌های پژوهشی بلندمدت اختصاص داده می‌شود.

معادل ۰٫۲۶ برابر از کل مقدار فروخته‌شده، به‌صورت سالانه و برای همیشه، به استخراج‌کنندگان تخصیص خواهد یافت.

تفکیک صدور (Issuance Breakdown)

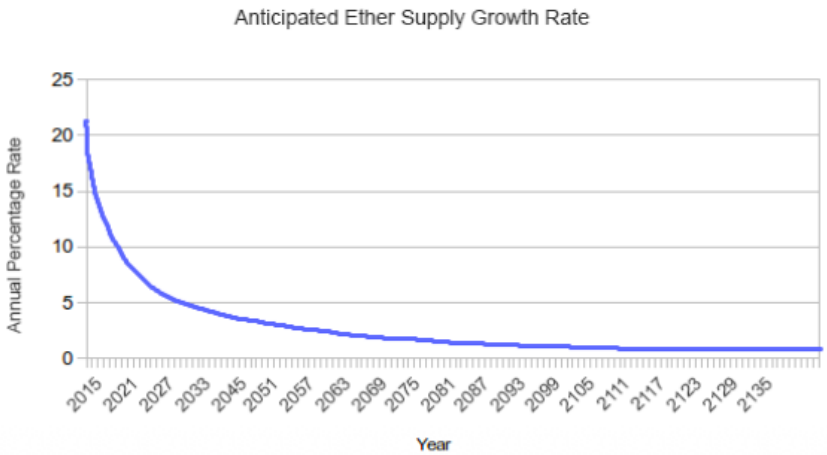
مدل رشد خطی و دائمی عرضه، ریسک آنچه برخی آن را تمرکز بیش‌ازحد ثروت در بیت‌کوین می‌دانند کاهش می‌دهد و به افرادی که در زمان حال و آینده زندگی می‌کنند، فرصت منصفانه‌ای برای به‌دست‌آوردن واحدهای پولی می‌دهد. در عین حال، این مدل از کاهش ارزش اتر جلوگیری می‌کند؛ زیرا «نرخ رشد عرضه» به‌عنوان یک درصد، با گذشت زمان به سمت صفر میل می‌کند.

همچنین این فرض را مطرح می‌کنیم که از آنجا که همواره مقداری از سکه‌ها به‌دلیل سهل‌انگاری، مرگ، از دست‌رفتن کلیدها و موارد مشابه از چرخه خارج می‌شوند، و می‌توان این پدیده را به‌صورت درصدی از کل عرضه در هر سال مدل‌سازی کرد، در نهایت مقدار کل عرضهٔ در گردش به یک مقدار پایدار خواهد رسید. این مقدار برابر است با نرخ صدور سالانه تقسیم بر نرخ از دست‌رفتن سکه‌ها.

برای مثال، اگر نرخ از دست‌رفتن سکه‌ها ۱٪ باشد، زمانی که عرضه به 26 Xبرسد، در هر سال 0.26 Xواحد جدید استخراج می‌شود و هم‌زمان 0.26 Xواحد نیز از بین می‌رود؛ در نتیجه، یک حالت تعادل پایدار ایجاد خواهد شد.

Group	At launch	After 1 year	After 5 years
Currency units	1.198X	1.458X	2.498X
Purchasers	83.5%	68.6%	40.0%
Early contributor distribution	8.26%	6.79%	3.96%
Long-term endowment	8.26%	6.79%	3.96%
Miners	0%	17.8%	52.0%

با وجود صدور خطی واحدهای پولی، همانند بیت‌کوین، نرخ رشد عرضه در طول زمان همچنان به سمت صفر میل می‌کند.



تمرکز در استخراج (Mining Centralization)

الگوریتم استخراج بیت‌کوین اساساً به این صورت عمل می‌کند که استخراج‌کنندگان، هش SHA256 را میلیون‌ها بار روی نسخه‌های اندکی تغییریافته‌ای از سرآیند بلاک محاسبه می‌کنند، تا در نهایت یکی از گره‌ها به نسخه‌ای برسد که مقدار هش آن کمتر از مقدار هدف باشد (که در حال حاضر حدود 190^2 است). با این حال، این الگوریتم استخراج در برابر دو نوع تمرکز آسیب‌پذیر است.

نخست، اکوسیستم استخراج به‌تدریج تحت سلطهٔASIC ها (Application-Specific Integrated Circuits) قرار گرفته است؛ تراشه‌هایی که به‌طور خاص برای استخراج بیت‌کوین طراحی شده‌اند و هزاران برابر کارآمدتر از سخت‌افزارهای عمومی در این وظیفه هستند. این مسئله باعث شده استخراج بیت‌کوین دیگر یک فعالیت غیرمتمرکز و برابر نباشد و مشارکت مؤثر در آن نیازمند سرمایه‌ای در حد میلیون‌ها دلار شود.

دوم، بیشتر استخراج‌کنندگان بیت‌کوین در عمل فرآیند اعتبارسنجی بلاک را به‌صورت محلی انجام نمی‌دهند و به‌جای آن به استخرهای استخراج متمرکز متکی هستند تا سرآیند بلاک‌ها را در اختیارشان قرار دهند. این مشکل حتی جدی‌تر است: در زمان نگارش این متن، دو استخر استخراج بزرگ به‌طور غیرمستقیم حدود ۵۰٪ از توان پردازشی شبکهٔ بیت‌کوین را کنترل می‌کنند. البته این خطر تا حدی با این واقعیت کاهش می‌یابد که استخراج‌کنندگان می‌توانند در صورت تلاش یک استخر یا ائتلاف برای انجام حملهٔ ۵۱٪، به استخرهای دیگر مهاجرت کنند.

هدف فعلی اتریوم این است که از الگوریتم استخراجی استفاده کند که بر پایهٔ تولید تصادفی یک تابع هش منحصربه‌فرد برای هر ۱۰۰۰ نانس باشد و از دامنه‌ای به‌اندازهٔ کافی گسترده از محاسبات استفاده کند تا مزیت سخت‌افزارهای تخصصی از بین برود. چنین راهبردی قطعاً تمرکز را به صفر نمی‌رساند، اما نیازی هم به این کار نیست.

توجه داشته باشید که هر کاربر عادی، با لپ‌تاپ یا رایانهٔ شخصی خود، می‌تواند مقدار مشخصی استخراج را تقریباً به‌صورت رایگان (صرفاً با پرداخت هزینهٔ برق) انجام دهد. اما پس از رسیدن به استفادهٔ ۱۰۰٪ از توان پردازشی سیستم، استخراج بیشتر مستلزم پرداخت هزینهٔ هم‌زمان برای برق و سخت‌افزار خواهد بود. در مقابل، شرکت‌های استخراج مبتنی بر ASIC از همان اولین هش، ناچار به پرداخت هر دو هزینهٔ برق و سخت‌افزار هستند. بنابراین، اگر میزان سود تمرکز کمتر از نسبت (E + H) / E باقی بماند، حتی در صورت تولیدASIC ها نیز همچنان برای استخراج‌کنندگان عادی فضا وجود خواهد داشت.

علاوه بر این، قصد داریم الگوریتم استخراج را به‌گونه‌ای طراحی کنیم که نیازمند دسترسی به کل بلاکچین باشد؛ امری که استخراج‌کنندگان را وادار می‌کند کل بلاکچین را ذخیره کرده و دست‌کم قادر به اعتبارسنجی تمام تراکنش‌ها باشند. این کار نیاز به استخرهای استخراج متمرکز را از بین می‌برد. هرچند استخرها همچنان می‌توانند نقش مشروع هموارسازی تصادفی بودن توزیع پاداش‌ها را ایفا کنند، اما این وظیفه به همان اندازه می‌تواند توسط استخرهای همتابه‌همتا بدون کنترل مرکزی انجام شود.

این رویکرد همچنین به مقابله با تمرکز کمک می‌کند، زیرا تعداد گره‌های کامل شبکه را افزایش می‌دهد و باعث می‌شود حتی اگر بیشتر کاربران عادی ترجیح دهند از کلاینت‌های سبک استفاده کنند، شبکه همچنان به‌اندازهٔ کافی غیرمتمرکز باقی بماند.

یکی از نگرانی‌های رایج دربارهٔ اتریوم، مسئلهٔ مقیاس‌پذیری است. همانند بیت‌کوین، اتریوم نیز از این ضعف رنج می‌برد که هر تراکنش باید توسط تمام گره‌های شبکه پردازش شود. در بیت‌کوین، اندازهٔ فعلی بلاکچین حدود ۲۰ گیگابایت است و با نرخی در حدود ۱ مگابایت در ساعت رشد می‌کند. اگر شبکهٔ بیت‌کوین بخواهد ۲۰۰۰ تراکنش در ثانیهٔ ویزا را پردازش کند، اندازهٔ بلاکچین آن به ازای هر سه ثانیه ۱ مگابایت (۱ گیگابایت در ساعت و ۸ ترابایت در سال) افزایش خواهد یافت.

اتریوم احتمالاً با الگوی رشد مشابهی مواجه می‌شود؛ حتی شدیدتر از بیت‌کوین، زیرا به‌جای یک کاربرد واحد (پول)، میزبان تعداد زیادی برنامهٔ مختلف خواهد بود. با این حال، این مشکل تا حدی با این واقعیت تعدیل می‌شود که گره‌های کامل اتریوم تنها نیاز دارند «وضعیت (state) «را ذخیره کنند، نه کل تاریخچهٔ بلاکچین.

مشکل بلاکچین‌های بسیار بزرگ، خطر تمرکز است. اگر اندازهٔ بلاکچین به‌عنوان مثال به ۱۰۰ ترابایت برسد، سناریوی محتمل این است که تنها تعداد اندکی از کسب‌وکارهای بزرگ گرهٔ کامل اجرا کنند و کاربران عادی صرفاً از گره‌های سبک SPV استفاده کنند. در چنین وضعیتی، این نگرانی به‌وجود می‌آید که گره‌های کامل بتوانند با یکدیگر تبانی کرده و به‌صورت سودآور تقلب کنند (برای مثال، پاداش بلاک را تغییر دهند یا به خودشان بیت‌کوین بدهند). گره‌های سبک راهی برای تشخیص فوری این تقلب نخواهند داشت.

البته احتمالاً دست‌کم یک گرهٔ کامل صادق وجود خواهد داشت و پس از چند ساعت، اطلاعات مربوط به تقلب از طریق کانال‌هایی مانند Redditمنتشر می‌شود، اما در آن نقطه دیگر خیلی دیر است. کاربران عادی ناچار خواهند بود تلاش بزرگی برای مسدودسازی بلاک‌های متقلبانه سازمان‌دهی کنند؛ کاری که از نظر هماهنگی، در مقیاسی مشابه یک حملهٔ موفق ۵۱٪ است و احتمالاً عملی نیست. در مورد بیت‌کوین، این مسئله در حال حاضر وجود دارد، اما یک اصلاح در بلاکچین که توسط پیتر تاد (Peter Todd) پیشنهاد شده می‌تواند این مشکل را کاهش دهد.

در کوتاه‌مدت، اتریوم از دو راهبرد اضافی برای مقابله با این مشکل استفاده خواهد کرد. نخست، به‌دلیل الگوریتم‌های استخراج مبتنی بر بلاکچین، هر استخراج‌کننده ناچار خواهد بود یک گرهٔ کامل باشد؛ موضوعی که یک حداقل تضمین‌شده برای تعداد گره‌های کامل ایجاد می‌کند. دوم و مهم‌تر از آن، اتریوم پس از پردازش هر تراکنش، ریشهٔ یک درخت وضعیت میانی را در بلاکچین ثبت خواهد کرد.

حتی اگر اعتبارسنجی بلاک‌ها متمرکز شود، تا زمانی که تنها یک گرهٔ اعتبارسنج صادق وجود داشته باشد، می‌توان از طریق یک پروتکل راستی‌آزمایی، مشکل تمرکز را دور زد. اگر یک استخراج‌کننده بلاک نامعتبری منتشر کند، آن بلاک یا از نظر ساختاری نادرست است، یا وضعیت نهایی S[n] اشتباه است. از آنجا که S[0] قطعاً درست است، باید نخستین وضعیت نادرست S[i] وجود داشته باشد که در آن S[i-1]درست بوده است. گرهٔ اعتبارسنج می‌تواند شاخص i را به‌همراه یک «اثبات نامعتبر بودن» ارائه کند که شامل زیرمجموعه‌ای از گره‌های درخت پاتریشیا است که برای اجرای APPLY(S[i-1], TX[i]) → S[i] لازم هستند. سایر گره‌ها می‌توانند با استفاده از این داده‌ها آن بخش از محاسبه را اجرا کرده و ببینند که S[i] تولیدشده با S[i] ارائه‌شده تطابق ندارد.

یک حملهٔ پیشرفته‌تر شامل این است که استخراج‌کنندگان مخرب بلاک‌های ناقص منتشر کنند، به‌گونه‌ای که اطلاعات کامل برای تشخیص اعتبار بلاک‌ها در دسترس نباشد. راه‌حل این مسئله، یک پروتکل چالش–پاسخ است: گره‌های اعتبارسنج «چالش»هایی در قالب شاخص تراکنش‌ها صادر می‌کنند و تا زمانی که یک گره (چه استخراج‌کننده و چه یک اعتبارسنج دیگر) زیرمجموعه‌ای از گره‌های پاتریشیا را به‌عنوان اثبات اعتبار ارائه نکند، گرهٔ سبک بلاک را غیرقابل‌اعتماد در نظر می‌گیرد.

کنار هم قرار دادن همه‌چیز: برنامه‌های غیرمتمرکز

سازوکار قراردادهایی که پیش‌تر توضیح داده شد، به هر کسی اجازه می‌دهد چیزی شبیه به یک برنامهٔ خط فرمان بسازد که روی یک ماشین مجازی اجرا می‌شود و از طریق اجماع در سراسر شبکه عمل می‌کند؛ برنامه‌ای که می‌تواند یک وضعیت سراسری و در دسترس همگان را به‌عنوان «هارد دیسک» خود تغییر دهد. با این حال، برای بیشتر کاربران، رابط خط فرمانی که همان سازوکار ارسال تراکنش است، به‌اندازهٔ کافی کاربرپسند نیست تا غیرمتمرکزسازی را به گزینه‌ای جذاب برای استفادهٔ عمومی تبدیل کند.

به همین دلیل، یک «برنامهٔ غیرمتمرکز» کامل باید از دو بخش تشکیل شود:

نخست، مؤلفه‌های سطح پایین منطق کسب‌وکار که می‌توانند به‌طور کامل روی اتریوم پیاده‌سازی شوند، یا ترکیبی از اتریوم و سیستم‌های دیگر باشند (برای مثال یک لایهٔ پیام‌رسانی همتابه‌همتا که قرار است در آینده در کلاینت‌های اتریوم تعبیه شود)، یا حتی کاملاً بر پایهٔ سیستم‌های دیگر عمل کنند؛

و دوم، مؤلفه‌های سطح بالای رابط کاربری گرافیکی.

طراحی کلاینت اتریوم به‌گونه‌ای است که نقش یک مرورگر وب را ایفا کند، با این تفاوت که از یک شیء API جاوااسکریپتی به نام «eth» پشتیبانی می‌کند. صفحات وب تخصصی که در این کلاینت نمایش داده می‌شوند می‌توانند از این API برای تعامل مستقیم با بلاکچین اتریوم استفاده کنند. از دید وب «سنتی»، این صفحات کاملاً محتوای ایستا هستند، زیرا بلاکچین و سایر پروتکل‌های غیرمتمرکز به‌طور کامل جایگزین سرور برای رسیدگی به درخواست‌های کاربران می‌شوند.

در نهایت، این امکان وجود دارد که خود صفحات وب نیز توسط پروتکل‌های غیرمتمرکز ذخیره شوند؛ پروتکل‌هایی که امید می‌رود به‌نوعی بر پایهٔ اتریوم یا در تعامل با آن ساخته شوند.

جمع‌بندی (Conclusion)

پروتکل اتریوم در ابتدا به‌عنوان نسخه‌ای ارتقایافته از یک ارز دیجیتال طراحی شد؛ نسخه‌ای که با تکیه بر یک زبان برنامه‌نویسی بسیار عمومی، قابلیت‌هایی پیشرفته مانند امانت‌داری درون‌زنجیره‌ای، محدودیت برداشت، قراردادهای مالی، بازارهای قمار و موارد مشابه را فراهم می‌کرد. پروتکل اتریوم به‌طور مستقیم از هیچ‌یک از این کاربردها «پشتیبانی» نمی‌کند، اما وجود یک زبان برنامه‌نویسی تورینگ‌کامل به این معناست که از نظر نظری می‌توان برای هر نوع تراکنش یا کاربردی، قراردادهای دلخواه ایجاد کرد.

با این حال، آنچه اتریوم را واقعاً متمایز می‌کند این است که فراتر از مفهوم صرفِ پول حرکت می‌کند. پروتکل‌ها و برنامه‌های غیرمتمرکز در حوزه‌هایی مانند ذخیره‌سازی غیرمتمرکز فایل، محاسبات غیرمتمرکز و بازارهای پیش‌بینی غیرمتمرکز — در کنار ده‌ها مفهوم مشابه دیگر — این ظرفیت را دارند که کارایی صنعت محاسبات را به‌طور چشمگیری افزایش دهند و با افزودن یک لایهٔ اقتصادی، جان تازه‌ای به سایر پروتکل‌های همتابه‌همتا ببخشند. افزون بر این، طیف قابل‌توجهی از کاربردها نیز وجود دارد که اساساً هیچ ارتباطی با پول ندارند.

ایدهٔ تابع انتقال وضعیت دلخواه، همان‌گونه که در پروتکل اتریوم پیاده‌سازی شده است، بستری با پتانسیلی منحصربه‌فرد ایجاد می‌کند. اتریوم، برخلاف پروتکل‌های بسته و تک‌منظوره‌ای که برای مجموعه‌ای محدود از کاربردها در ذخیره‌سازی داده، قمار یا امور مالی طراحی شده‌اند، ذاتاً یک پلتفرم باز و توسعه‌پذیر است. ما بر این باوریم که اتریوم به‌طرزی چشمگیر برای ایفای نقش به‌عنوان یک لایهٔ بنیادین برای طیف بسیار گسترده‌ای از پروتکل‌های مالی و غیرمالی در سال‌های آینده مناسب است.

یادداشت‌ها و منابع تکمیلی(Notes and Further Reading)

یادداشت‌ها

1. یک خوانندهٔ دقیق ممکن است متوجه شود که در واقع، آدرس بیت‌کوین هشِ کلید عمومی منحنی بیضوی (Elliptic Curve Public Key)است و نه خودِ کلید عمومی. با این حال، از نظر اصطلاحات رمزنگاری، اطلاق «کلید عمومی» به هشِ کلید عمومی کاملاً معتبر است. دلیل آن این است که رمزنگاری بیت‌کوین را می‌توان نوعی الگوریتم امضای دیجیتال سفارشی در نظر گرفت؛ جایی که کلید عمومی شامل هشِ کلید عمومی ECC است، امضا شامل خودِ کلید عمومی ECC به‌همراه امضای ECC بوده و الگوریتم اعتبارسنجی، ابتدا کلید عمومی ECC موجود در امضا را با هشِ ارائه‌شده به‌عنوان کلید عمومی تطبیق می‌دهد و سپس امضای ECC را با همان کلید عمومی بررسی می‌کند.
2. از نظر فنی، مقدار مورد استفاده، میانهٔ ۱۱ بلاک قبلی است.
3. در سطح داخلی سیستم، هر دو مقدار «۲» و «CHARLIE» عدد محسوب می‌شوند؛ با این تفاوت که مقدار دوم به‌صورت نمایش مبنای ۲۵۶ با ترتیب بایت big-endian ذخیره می‌شود. اعداد می‌توانند حداقل مقدار ۰ و حداکثر مقدار 2256-12^{256}-12256-1را داشته باشند.

منابع تکمیلی

1. ارزش ذاتی:(Intrinsic value)
https://tinyurl.com/BitcoinMag-IntrinsicValue
2. مالکیت هوشمند:(Smart property)
https://en.bitcoin.it/wiki/Smart_Property
3. قراردادهای هوشمند:(Smart contracts)
https://en.bitcoin.it/wiki/Contracts
4. b-money:
http://www.weidai.com/bmoney.txt
5. اثبات کار قابل‌استفادهٔ مجدد:(Reusable proofs of work)
http://www.finney.org/~hal/rpow/
6. عناوین مالکیت امن با اختیار مالک:(Secure property titles with owner authority)
http://szabo.best.vwh.net/securetitle.html
7. وایت‌پیپر بیت‌کوین:
http://bitcoin.org/bitcoin.pdf
8. نیم‌کوین:(Namecoin)
https://namecoin.org/
9. مثلث زوکو:(Zooko’s triangle)
http://en.wikipedia.org/wiki/Zooko's_triangle

10. وایت‌پیپر کوین‌های رنگی:(Colored coins)
https://tinyurl.com/coloredcoin-whitepaper

11. وایت‌پیپر مسترکوین:(Mastercoin)
https://github.com/mastercoin-MSC/spec

12. شرکت‌های خودگردان غیرمتمرکز، :Bitcoin Magazine
https://tinyurl.com/Bootstrapping-DACs

13. تأیید پرداخت ساده‌شده:(Simplified Payment Verification)
https://en.bitcoin.it/wiki/Scalability#Simplifiedpaymentverification

14. درخت‌های مرکل:(Merkle trees)
http://en.wikipedia.org/wiki/Merkle_tree

15. درخت‌های پاتریشیا:(Patricia trees)
http://en.wikipedia.org/wiki/Patricia_tree

16. پروتکل:GHOST
http://www.cs.huji.ac.il/~avivz/pubs/13/btc_scalability_full.pdf

17. StorJ و عامل‌های خودگردان، :Jeff Garzik
https://tinyurl.com/storj-agents

18. سخنرانی Mike Hearn دربارهٔ Smart Property در جشنوارهٔ:Turing
http://www.youtube.com/watch?v=Pu4PAMFPo5Y

19. RLP اتریوم:
https://github.com/ethereum/wiki/wiki/%5BEnglish%5D-RLP

20. درخت‌های مرکل پاتریشیا در اتریوم:
https://github.com/ethereum/wiki/wiki/%5BEnglish%5D-Patricia-Tree

21. Peter Todd دربارهٔ Merkle Sum Trees:
http://sourceforge.net/p/bitcoin/mailman/message/31709140/